A Semi-reliable Multicast Protocol for Distributed Multimedia Applications in Large Scale Networks

Christiane Montenegro Bortoleto[1], Lau Cheuk Lung[1], Frank A. Siqueira[2], Alysson Neves Bessani[3], Joni da Silva Fraga[3]

[1]Pontifícia Univ. Católica do Paraná,Programa de Pós-Graduação em Informática Aplicada
{cbortoleto, lau}@ppgia.pucpr.br

[2]Universidade Federal de Santa Catarina, Departamento de Informática e Estatística
frank@inf.ufsc.br

[3]Universidade Federal de Santa Catarina, Departamento de Automação e Sistemas
{neves, fraga}@das.ufsc.br

Abstract. This paper proposes a semi-reliable multicast protocol that aims to increase the quality of video streams transmitted in large-scale systems without overloading the video source and the communications network. This protocol, which is based on the IP multicast protocol and the MPEG standard, evaluates the necessity of retransmitting lost packets taking into account the capacity of the corresponding MPEG frames to improve the quality of the video stream. The proposed protocol relies on the neighboring receivers for retransmitting lost packets, resulting in much faster recovery, which is vital in order to receive retransmitted packets in time to be exhibited. Besides, this strategy avoids overloading the video source, making it more scalable than the traditional approach of retransmitting from the source.

1 Introduction

Real-time multimedia applications have been widely developed in many platforms and topologies. Internet and intranet structures employ more and more integrated networking services, where applications with different characteristics are executed using the same communication infrastructure and service models. However, the ordinary communication infrastructure is risky to the quality of multimedia content due to the implementation of transmission policies, routing algorithms, packet discard strategies and due to the complexity of media data formats. Networking infrastructures need mechanisms to improve and guarantee the performance of these applications. Physical limitations - e.g. router overload caused by directing and retransmitting datagrams - may lead to the loss of information required by real-time and multimedia applications, affecting negatively their behavior.

Distributed multimedia applications have singular requirements that are not found in other kinds of distributed applications. In videoconferences through the Internet, for example, the communication support is the main cause of low performance. An audio/video stream requires data to be received at the right moment and can handle some data loss. If a packet arrives too late it does not contribute to the exhibition, meaning that, for the viewer, the effect would be the same as if the packet had never arrived. An acceptable amount of data loss in the audio/video stream can be handled without causing a significant loss of quality noticeable for the viewer.

In the literature, there is a substantial amount of work on best-effort multicast for distributed multimedia applications and reliable multicast for applications that demand reliable message delivery (i.e. fault tolerant applications). Between those two approaches lie the semi-reliable multicast protocols, which represent a more recent category of group communication protocols that are still maturing [12; 9; 11; 13]. Semi-reliable multicast is a communication paradigm in which data are classified, usually by the application, before being transmitted, establishing different importance or priority levels for error recovery (retransmission).

This paper presents a semi-reliable multicast protocol which was designed to be more efficient and scalable than reliable protocols. This protocol was designed to be used by distributed multimedia applications based on groups (i.e. digital video multicast). Through simulations, it is shown that this protocol is more efficient for use in large scale networks. The protocol is specified for video streaming multicast applications which use encoders that provide some kind of frame hierarchy, such as the MPEG standard, allowing the establishment of priority levels for frame recovery.

It is important to emphasize that the proposed protocol is not intended to be employed as a complete multicast protocol for media transfer. In this paper, we propose and study the impact of a data recovery technique that can be combined with other protocols found in the literature in order to provide semi-reliable media delivery.

This paper presents in section 2 the main characteristics of the MPEG standard. Section 3 describes the concept of semi-reliable multicast. The proposed protocol is presented in section 4. In section 5, experimental data is shown. Section 6 presents related proposals and, finally, section 7 presents the final conclusions and perspectives for future work.

2 The MPEG Standard

The most widely adopted standards for video compression belong to the MPEG (Motion Picture Experts Group) family [5]. There are different standards, such as MPEG-2, for example, which requires transfer rates from 3 Mbps to 100 Mbps. The MPEG-2 compression algorithm is based on pixel correlation and translational movement correlation between consecutive frames. It takes into account that the pictures in an image sequence are very similar, except for disjoints due to movement, so it is possible do code a frame through calculating the movement vector related to the previous frame [13]. The output stream consists of three types of frames:

- I-frames (Intra-coded): complete images individually coded;
- P-frames (Predictive): coded frames with prediction related to the previous frame;
- B-frames (Bidirectional): differences between the previous and the next frames.

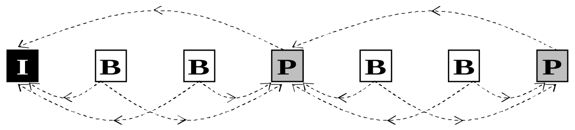

Fig. 1. IPB-frame Relation

The I frames are inserted in the output stream at a specific rate, with P and B frames between them. The I frames do not depend on other frames to be decoded, but they are necessary on P and B frame decoding. P-frames are needed on B frame decoding and they are based on forward prediction using the previous frame as reference, which can be a P or I frame. The P frame has a past movement reference vector, used as reference to the previous frame in the same position of the present one. B frames are based on backward prediction using the previous and the following frames, which can be I or P type. B frames have a forward movement vector, used as reference the next frame on the same position of the present one. This coding method makes some frames more important than others. If an I frame is lost during transmission, it will not be possible to decode the B and P frames that arrive before the next I frame. The relation between frames in an MPEG stream is shown in Fig. 1. Each GOP (Group Of Pictures) contains one I frame. A GOP is a set of frames where each one has the picture header and its present data. The GOP sequence is an N-sized sequence of frames between two consecutive I frames.

3 Semi-reliable Multicast

Semi-reliable multicast is a communication abstraction where not every packet is necessarily retransmitted when requested; only the most important packets for the application are given higher priority to be recovered. It means that the reliable delivery for a given set of packets which will be sent in a receiver group is granted only to a subset (i.e., the packets with higher priority). Reliable delivery means that a multicast packet has to be delivered to every correct receiver (agreement property [7]). This guarantee can be obtained through an error correction mechanism (i.e. packet retransmission). For other packets (with lower priority), transmission errors will only be corrected if the network conditions allow it. If the conditions are not favorable, the delivery is based on best-effort, where the agreement property can be infringed.

Packets to be multicast can be classified according to a hierarchy, based on some application semantic [11] or some rule from the semi-reliable multicast protocol itself, establishing importance or priority levels for error correction (selective retransmission [12, 9]). Error correction is made according to this property and to network state parameters (i.e. traffic, congestion, delay, and so on).

3.1 Semi-reliable Multicast for Distributed Multimedia Applications

Multimedia information transmission has a fixed rate and frames have to be received and rendered in the receiver with a similar rate to keep the original meaning of the sequence. Thus, each unit of information sent must be received within a certain time bound. Besides, the data loss ratio (which includes data delivered after the moment it was supposed to be rendered) should also be kept within the boundaries defined by the application. So, the quality of service (QoS) issue includes finding the boundaries through the networks for transmission errors and jitter.

A traditional reliable multicast tool is not appropriate for multimedia multicast for many reasons. The retransmission strategy with timeout achieves reliability through latency increase. Multimedia applications can tolerate errors due to lost and corrupted packets as long as they are kept within an acceptable limit. Thus, a multimedia transport protocol demands semi-reliable delivery where delay is more relevant than delivering every packet of the set. Other kinds of applications can use semi-reliable multicast for performance improvement. In [11], the author describes distributed multiplayer games as an example where obsolete messages can be eliminated without damaging the final result.

4 The Proposed Protocol

The proposed protocol uses the frame hierarchy defined in the MPEG standard for semi-reliable multicast through classifying MPEG frames and encapsulating them in UDP packets. Once having the frames classified, error correction (lost packets retransmission) can be made by the sender (source) or by the receivers, according to the lost frame type (I, P or B) and to some network parameters (traffic, reception rate, congestion, etc). The idea is that every I frame is reliably delivered and P and B frames, specially the first one, are retransmitted (if a loss occurs) depending on the conditions of the transmission environment. The proposed protocol is based on ideas from ReMIOP [4] and some concepts presented in [10].

In order to develop this protocol, the following assumptions were made:

§ The environment presents no guarantees for message delivery.
§ Neither the source nor the receivers know the members of the multicast group.
§ The source sends one frame (I, P or B) per packet.
§ Each packet carries information about: the sequence number of the frame it contains, the type of frame it contains and the type of the last eight frames sent.
§ Both source and receivers are able to send messages to and receive messages from the multicast group.

The algorithm presented next describes the procedure to receive and send messages using the proposed protocol. In this algorithm it is assumed that when the sender detects a lost packet it is able to know what kind of MPEG frame it was carrying.

4.1 Protocol Description

Frames (packets) are multicast directly to the group, without previous knowledge of its members. Receivers put all received packets in a buffer and deliver them to the application (lines 5-8). In line 6 the receiver cancels any possible retransmission request for the received message. Then, receivers detect a packet loss (line 12) through the search for blanks in the sequence of received packets, which is expressed by a sequence number. When a loss is detected, the receiver evaluates if it is necessary to request retransmission through a NACK message (lines 13, 18 and 19). This evaluation is based on application QoS needs and network parameters. Application QoS considers relevance of the lost frames for media reproduction in the receiver and the usability of the lost frame by the time the retransmitted packet arrives at the receiver. Network parameters consider packet loss rate, congestion and acceptable delay.

If a lost frame is considered relevant (lines 16-19) retransmission is requested through a NACK message which is multicast to the group (line 21). If the is not relevant, the receiver ignores the lost packet. Any process (sender or receiver) which receives a NACK and has the requested frame, evaluates again the parameters (lines 33 and 36) and, if that is the case, multicasts it again to the group (line 38). Note that this QoS parameter evaluation for selective retransmission is made just for packets containing P and B-frames. Lost packets containing I frames are always retransmitted (lines 13-15 and 30-32).

```
1. //RECEIVER’S ALGORITHM
2. WHEN receives(m)
3.    IF m.type = DATA {if receives a data message}
4.      frame = m.type_of_frame
5.       IF search_buffer(m.sender, m) == NULL
6.         cancels_booked(NACKm) {cancels NACK for m}
7.         adds_to_buffer(m.sender, m)
8.         delivers(m) {delivers the message}
9.       ELSE
10.        cancels_booked(m) {cancels m retransmission}
11.      END-IF
12.      IF there is a missing message {recovery}
13.         IF frame == “I”
14.            wait(random(Tnack))
15.                       multicast(NACKm)
16.           ELSE IF still_relevant(frame)
17.             loss_rate = get_loss_rate()
18.            IF (frame == “P” AND loss_rate < 40%) OR
19.               (frame == “B” AND loss_rate < 20%)
20.                        wait(random(Tnack))
21.                          multicast(NACKm)
22.             END-IF
23.          END-IF
24.       END-IF
25.   ELSE IF m.type == NACK {retransmission request}
26.      FOR EVERY mn ∈ m.nacked
27.         cancels_booked(NACKmn) {cancels NACK for mn}
28.         mT = search_buffer(m.sender, mn)
29.         IF mT ≠ NULL
30.            IF mT.frame == “I”
31.               wait(random(Trepair))
32.               multicast(mT)
33.                     ELSE
34.                        loss_rate = get_loss_rate()
35.               IF(frame ==“P” AND loss_rate< 40%) OR
36.                  (frame ==“B” AND loss_rate< 20%)
37.                  wait(random(Trepair))
38.                  multicast(mT)
39.                   END-IF
40.             END-IF
41.          END-IF
42.       END-FOR
43.    END-IF
44. END-WHEN
```

4.2 Retransmissions

The proposed protocol is based on the principle of retransmission made by the receiver, where the receivers share the responsibility of helping their peers to recover their losses [6]. To analyze the decision parameters, it was defined that the maximum loss rate to require retransmission or to retransmit packets is 20% for packets containing B-frames and 40% for packets containing P-frames (lines 8-19 and 35-36).

The loss rate at each receiver is calculated based on a sample sequence – a window with the N last multicast packets. The protocol counts, for every window, the number of missing packets based on the sequence numbers. So, the get_loss_rate function (lines 17 and 34) calculates de percentage of lost packets for every N multicast packets.

The frame relevance is verified by the still_relevant function (line 16), which checks if the missing frame is still valid, i.e., if the moment for the packet to be shown by the application has not passed yet.

To avoid NACK or retransmission explosions the protocol employs a wait function that interrupts the execution during a random interval (whose superior limits are Tnack and Trepair respectively). When a receiver Ri is about to send a NACK, it waits for a random time Tnack (lines 14 and 20). If during this period Ri receives a NACK from another receiver Rj requesting the same packet, Ri cancels its NACK (line 27). In a similar way, when Ri receives a NACK and has the requested packet, it waits for a random time Trepair before multicasting this packet (lines 32 and 38). But if within this period Ri receives the requested packet, Ri cancels the retransmission (line 10).

4.3 Exhibition Buffer and Retransmission Buffer

An important issue related to the implementation of the proposed protocol is the exhibition buffer management in the receiver to deal appropriately with indirect losses, i.e. the discard of B and P frames that occurs due to the loss of an I-frame. Two classes were created to store received packets: WaitElement and WaitWindow. The WaitElement class represents a set of received packets where the first one contains an I-frame - i.e. a GOP - preceded by two control fields: the I-frame sequence number of that set and a flag that indicates if the I-frame is in the auxiliary buffer that stores the packets of that set. Fig. 2 illustrates this class.

The WaitWindow class represents a buffer that stores WaitElement objects. Packets that get to the receiver are put in the corresponding WaitElement, according to their sequence numbers. WaitElement objects are stored in the WaitWindow of the receiver, ordered by the sequence number of the corresponding I-Frame. Fig. 3 shows an example of WaitWindow object for a given frame sequence with some missing frames.

For each received packet containing an I-frame, a new instance of WaitElement is created and added to the WaitWindow object in the correct position, according to its sequence number. For packets containing P or B-frames, the WaitWindow object is searched for the WaitElement containing the I-frame that precedes the present frame. When it is found, the packet is added to the auxiliary buffer in the right position.

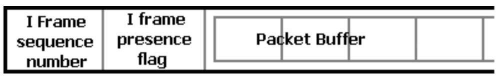

Fig. 2. WaitElement Class

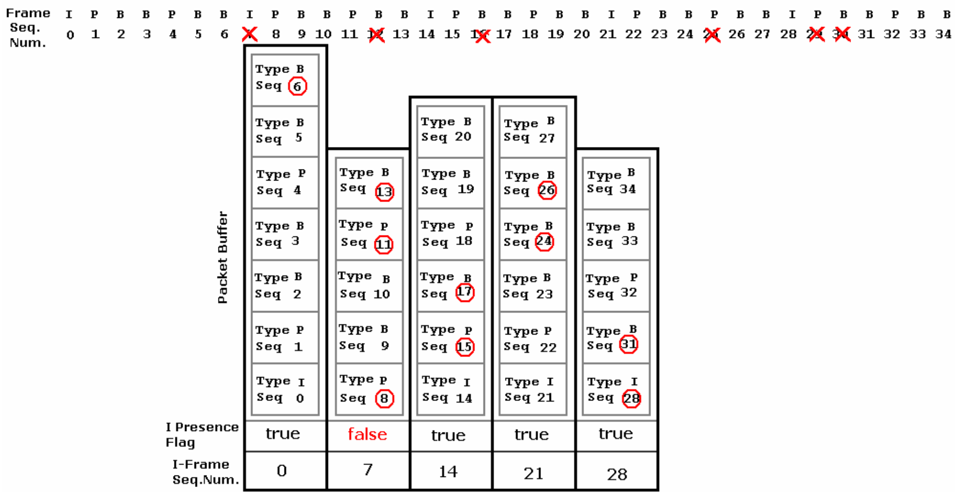

Fig. 3. WaitWindow Object

If the loss of a packet containing an I-frame is detected, an instance of WaitElement is created with the sequence number of the missing packet and then its flag is set to false, indicating that the I-frame is not inside the auxiliary buffer. Thus, if it comes the time to show that set of frames before the I-frame arrives, the whole set is discarded and the next set containing an I-frame goes to the exhibition buffer.

This algorithm allows that by the time the frames are sent to the exhibition buffer they have already been reordered and that discards have been made. Retransmission also takes place employing the frames kept in the WaitWindow object, occurring until the time the frames are sent to the application for being rendered. Using this structure, each receiver must have one entity to take care of the frame sending to the application. This is done by passing the frames from the WaitWindow to the exhibition buffer, which will be read by the application.

Each receiver has a retransmission buffer that stores the last received packets (frames). The packets in the exhibition buffer are also used by the receiver to answer retransmission requests (NACKs). This buffer has limited size and older packets are discarded when the limit is achieved by the FIFO (First In First Out) method.

5 Simulations and Results

In order to evaluate the proposed protocol, we have simulated its behavior using the Simmcast [3] network simulator. In these testes, comparisons were made between the proposed protocol, an ordinary multicast protocol, a NACK-sending multicast protocol and a reliable multicast protocol in terms of error correction, recovering time, receiver overload and video quality factor [10]. The topology adopted for the simulations, which was chosen with the intent of showing the adequacy of the proposed protocol for large scale networks, is illustrated by Fig. 4.

Simulations were performed using the same conditions for all four protocols. The adopted parameters are (section 4.2): N=50, Trepair = Tnack = 200ms.

In terms of lost packets recovery, the proposed protocol has presented an average of 81.7% of recovery against 2% of the multicast with NACK (simple retransmission) and 89% of the reliable multicast. The best performance from the proposed protocol and the reliable multicast can be credited, among other features, to the receiver-based retransmission. The reliable multicast has achieved a better rate because it tries to recover every single packet, no matter the temporal relevance of the packet for the application. The recovering averages for each type of frame are shown in Table 1.

It is convenient to compare the performance of the proposed protocol with the performance of the reliable multicast protocol in terms of discards. The results are shown in Table 2. Based on these figures, it is possible to notice that the reliable multicast protocol recovers more frames than the proposed protocol but great part of recovered data is discarded. It means that the proposed protocol is more efficient, once the recovered packets are more often useful to the application by the time they arrive.

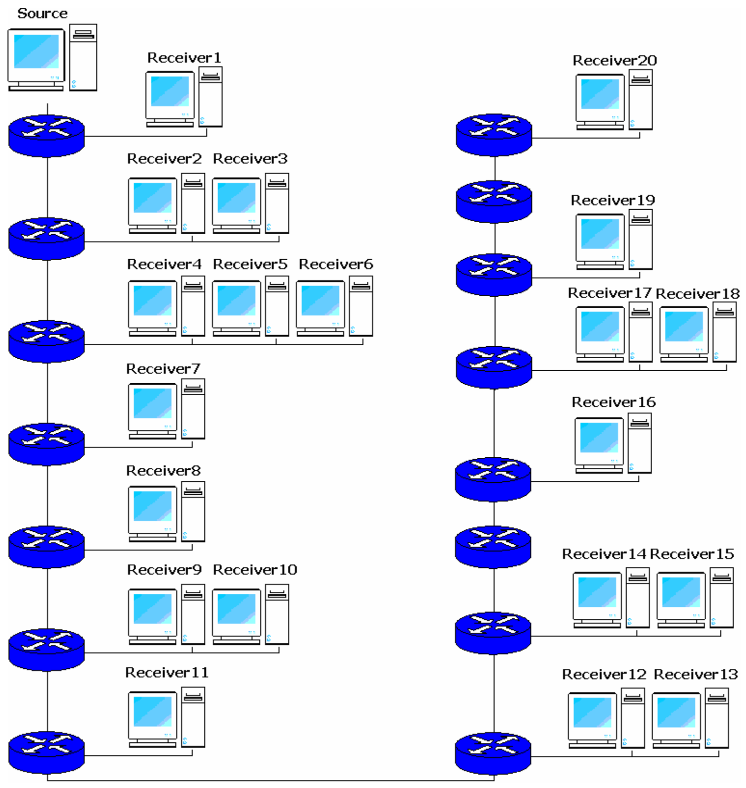

Fig. 4. Topology used in simulations

Table 1. Recovery per frame type

	% of recovered frames		
	I	**P**	**B**
Proposed Protocol	81.7	82.9	61.7
Multicast with NACK	1.0	2.1	3.4
Reliable Multicast	97.0	100.0	71.0

Table 2. Discards

	% of discarded frames		
	I	**P**	**B**
Proposed Protocol	45,4	33,0	24,8
Reliable Multicast	78,0	50,6	48,1

Another important result observed in the simulation is that the majority (93%) of recovered packets for the proposed protocol come from other receivers, instead of the source. This causes an improved performance in recovering time. For the topology shown in Fig. 4, the proposed protocol had an average recovery time of 95ms against 140ms of the reliable multicast and 962ms of the simple retransmission multicast.

A very satisfactory result obtained with the proposed protocol was the NACK suppression: an average of 39% of the NACKs were suppressed. The results show that the rate increases when the receiver is more distant from the source, as shown in Fig.5. This happens because the receivers that are closer to the source detect the losses more quickly and send their NACKs before the more distant receivers.

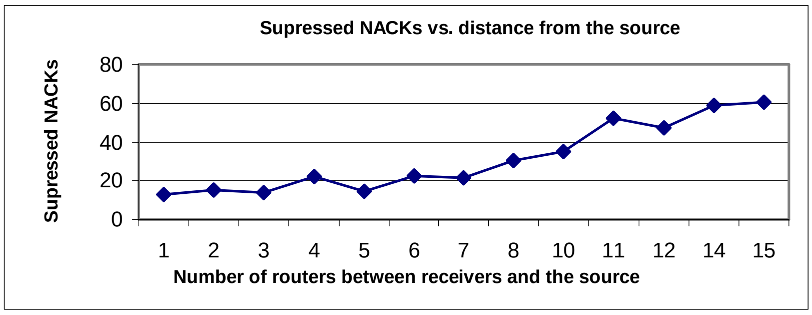

Fig. 5. Suppressed NACKs vs. distance from the source

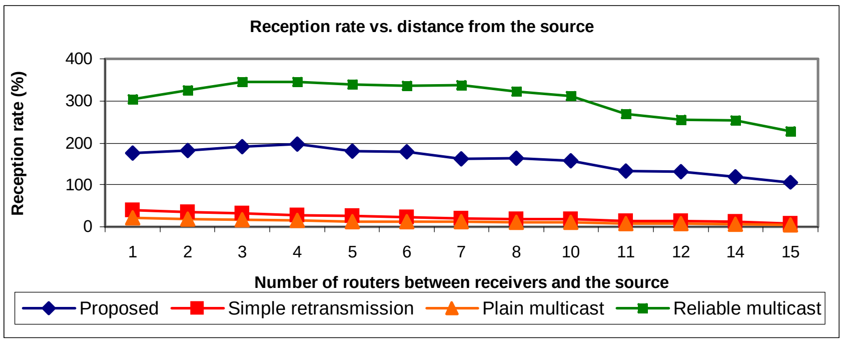

Fig. 6. Reception rate

The reception rate obtained by the proposed protocol is shown is Fig.6. This feature varies according to the distance from the source.

Another parameter used in the evaluation was the Video Quality Factor (q), defined in [10]. This parameter consists in a metric based on the GOP structure to evaluate video quality. The formula, showed in equations 1 and 2, takes into account direct and indirect losses. Direct losses are the ones caused by losing the frame itself. Indirect losses are caused by the loss of another frame; for example, a P or B frame which cannot be exhibited because there is an I frame missing. It is important to highlight that this metric claims to evaluate video flow information transport, not video quality from the point of view of the observer.

$$q = \frac{a_I * x_I + a_P * x_P + a_B * x_B}{a_I * N_{TI} + a_P * N_{TP} + a_B * N_{TB}}, 0 \leq q \leq 1 \qquad \textbf{(1)}$$

where:

j:	Represents the frame type (I, P or B)
x_j:	Represents the number of *j*-frames received and exhibited
N_{Tj}:	Represents the total number of *j*-frames in a GOP
a_j:	Represents the relative coefficient (*j*-frames in the GOP)

and

$$a_j = \frac{N_{Ij}}{N_{II} * N_{TI} + N_{IP} * N_{TP} + N_{IB} * N_{TB}} \qquad \textbf{(2)}$$

where

N_{Ij}:	Represents direct and indirect losses caused by the loss of the *j*-frame.

The proposed protocol, as shown in Fig.7, has presented a better performance compared to all the other tested protocols, despite not recovering as many frames as the reliable multicast protocol. This result was obtained due to the selective discard of packets containing less relevant frames.

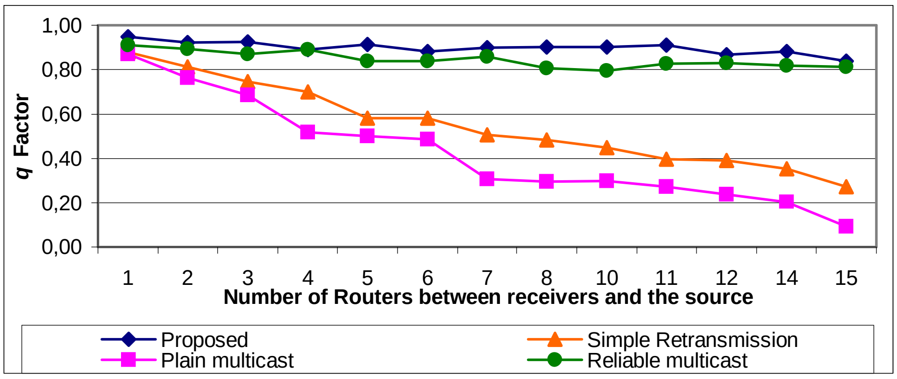

Figure 7. q Factor vs. distance from the source

6 Related Work

It is possible to find in the technical literature papers proposing a few semi-reliable multicast protocols. The WAIT protocol [9], for example, is designed to adjust itself to different quality requirements of a multimedia session and to reduce the network load, providing an improved quality of service for applications exchanging data through the Internet without relying on routers to do so. The problem with this approach is that, in order to form a group, the receivers must have information about the topology of the network, which reduces the scalability of the protocol.

SRP [12] (Selective Retransmission Protocol) uses a specific decision algorithm for each application to determine if a retransmission request for a lost packet should be answered or not, adjusting the loss and latency levels according to the application. Just a percentage of the lost data is retransmitted. The amount of retransmission depends on QoS factors including total losses, latency RTT (round trip time), network congestion and quality required by the user.

An approach based on semantically reliable multicast protocols is presented in [11]. The proposed model for a reliable multicast protocol eliminates obsolete messages to sustain a higher throughput. The idea is that message obsolescence is only used to avoid network congestion. When the buffer occupancy gets beyond the established limit, the protocol searches for obsolete messages in the buffer and purges them. When the buffer occupancy gets back to normal, the protocol gets reliable again. Both source and receiver can purge obsolete messages from their local buffers. The results show that this protocol can improve the throughput stability even with limited performance receivers.

PRTP (Partially Reliable Transport Protocol) [13] presents a partially reliable service that does not insist on recovering every error. Instead, it recovers part of lost data and improves packet delivery, allowing applications to exchange a controlled amount of loss for better throughput. This implies that the application itself must define a minimum reliability. When the parameter level of reliability is above the limit, the receiver does not ask for retransmission and sends a positive ACK. If a packet is lost, the receiver checks if the reliability level is above the limit. If it is, the receiver sends an ACK.

Yavatkar and Manoj have proposed a quasi-reliable multicast transport protocol for transmitting multimedia information in large scale [14]. The authors state that, due to the nature of multimedia communication, the protocol must use forward error correction to avoid delays inherent to flow-based and error control techniques.

SRM (Scalable Reliable Multicast) [6] is a reliable multicast protocol in which the retransmission system of the proposed protocol is based. In this protocol, every time a loss is detected, a NACK is sent to the whole multicast group and any member having the packet can retransmit it. To avoid duplicated NACKs or packets, the node establishes a random time before sending them. If the node receives the packet or NACK it was about to send, it cancels the sending process.

PRMP (Polling-based Reliable Multicast Protocol) [1] is a reliable multicast protocol with a source-based recovery mechanism. This protocol tries to solve the problem of limited scalability through an election-based mechanism that avoids implosion. Receivers are chosen in carefully planned moments in a way that, despite different RTT sets, the feedback packet rate at the source does not exceed source or network capacity.

These protocols are complete solutions, with congestion and flow control mechanisms. The proposed protocol is a simpler idea which experiments the concepts of receiver-based retransmission combined with selective discards. In a near future, we intend to provide a complete protocol with congestion and flow control mechanisms.

7 Conclusion

This paper presented a protocol to improve video data delivery through the network. Based on MPEG standard and multicast technology, this protocol guarantees reliable delivery of all video frames or part of them. Its good performance is obtained by a receiver based retransmission method that uses selective discard of lost packets according to the relevance of MPEG frames contained in them.

In comparison to ordinary multicast and NACK sending multicast, it is possible to notice that the proposed protocol is more efficient in large scale networks and it has better results for receivers that are at a bigger distance from the source.

References

1. Barcellos, Marinho P. (1998). "PRMP: A Scaleable Polling-based Reliable Multicast protocol". Ph.D. Thesis, Dept. of Computing Science, Univ. of Newcastle, October 1998.
2. Marinho P. Barcellos, Valter Roesler (2000), "M&M: Multicast e Multimídia", SBC/JAI2000 - XX Congresso da Sociedade Brasileira de Computação - pp.203-244. July 2000.
3. Barcellos, Marinho et al. (2001). "Simmcast: a Simulation Tool for Multicast Protocol Evaluation". XI Simpósio Brasileiro de Redes de Computadores, Florianópolis, 2001.
4. Bessani, Alysson N., Lung, Lau C., Fraga, Joni da S. (2003). "ReMIOP: Design and Implementation of a reliable multicast mechanism in CORBA" technical report. LCMI-UFSC.
5. Chiariglione, Leonardo (2000). "Short MPEG-2 Description", April 2000. Available at http://mpeg.telecomitalialab.com/standards/mpeg-2/mpeg-2.htm.
6. Floyd, S. et al (1995) "A reliable multicast framework for light-weight sessions and application level framing". Proc. of the ACM SIGCOMM 95, August 1995, p.342-356
7. Hadzilacos, V. and Toueg, S. (1994). "A modular approach to the specification and implementation of fault-tolerant broadcasts". Technical report, Department of Computer Science, Cornell University, New York - USA.
8. Liao, T (1998). "Light-weight reliable multicast protocol". http://webcanal.inria.fr/lrmp/.
9. Mane, Pravin. (2000) "WAIT: Selective Loss Recovery For Multimedia Multicast", MSc Thesis - Computer Science Department, WPI. May 2000.
10. Martins, R. F. Leite, C. A., Farines, J-M (2003), "Toward Quality Evaluation and Improvement of a MPEG Vídeo Stream". Proc. of the 3th IEEE Latin American Network Operations and Management Symposium – LANOMS'03, Foz do Iguaçu, Brazil, 2003.
11. Pereira, Jose et al (2003) "Semantically Reliable Multicast: Definition, Implementation, and Performance Evaluation". IEEE Transactions on Computers, vol.52, no.2, p150-165, 2003.
12. Piecuch, Mike et al. (2000) "A Selective Retransmission Protocol for Multimedia on the Internet". Proc. of SPIE Int. Symp. on Multimedia Systems and Applications. Nov. 2000.
13. Schneyer, S., Garcia, J., Brunstrom, A., Asplund, K. (1999) PRTP: "A Partially Reliable Transport Protocol for Multimedia Applications", Proceedings Int. Symp. on Intelligent Multimedia and Distance Education (ISIMADE), Baden-Baden, Germany, August 1999.
14. Yavatkar, Rajendra e Manoj, Leelanivas (1993) "Optimistic Strategies for Large-Scale Dissemination of Multimedia Information", First of the ACM Int. Conf. on Multimedia '93.