AIAA-2002-1876

PERFORMANCE OF AN IP RELIABLE MULTICAST PROTOCOL OVER A GEO SATELLITE ATM LINK

M. P. Howarth and Z. Sun
Centre for Communication Systems Research,
University of Surrey, Guildford, Surrey GU2 7XH, UK
Email: m.howarth@surrey.ac.uk

Abstract[1]

Communications satellites seem highly suitable for the delivery of advanced multimedia applications to end-users. This paper describes an investigation of the behaviour of a reliable multicast protocol, simplified from the experimental Internet protocol PGM. Results are presented from a simulation of a file transfer application that uses this protocol over an IP/ATM satellite link subject to burst errors. Theoretical expressions for the traffic volumes are presented and good agreement is shown between simulation and theory.

1. Introduction

Demand continues to grow for the Internet to support advanced multimedia applications such as video and audio streaming, videoconferencing, and information distribution. Communications satellites have significant potential to support the global delivery of these services and, in particular, satellites seem highly suitable for multicast applications. As broadband satellites are developed with on-board processing and switching, the study of IP multicast over networks that include satellite links consequently becomes of significant interest.

The potential advantages of satellite-based IP services have been well-rehearsed [1], and include their potentially global reach, cost-effective broadcast facilities, easy addition of new users with equipment installation only required at the customer's premises, alternative protection paths for existing network connections and ability to assign bandwidth on demand.

Geostationary earth orbit (GEO) communications satellites appear to offer advantages over low earth orbit (LEO) systems. The drawbacks of the latter include technical issues such as handoff requirements and transmission delays caused by inter-satellite links, together with commercial factors such as the cost, implementation timescales and maintenance effort of supporting a large constellation. However, GEO satellites have drawbacks of their own including the high round-trip time, limited transmission power and data rates, and the error characteristics of the satellite link.

Broadband satellites are currently typically being developed with either an ATM or DVB-S link layer, and this paper considers the former. An example architecture for an IP / ATM satellite system is described in detail by Yegenoglu [11], where an ATM switch is located on board the satellite, with IP routing and IP/ATM address translation controlled by a ground based "satellite route server". GEOCAST [3] is a satellite system designed to support multicast transmission. This system, probably based on ATM, has two principal scenarios:

- The satellite acting as a transit provider in an ISP's edge network, located between the Internet backbone and the ISP's PoP (point of presence);
- The satellite providing connectivity between end users and ISPs, who are located in either the same or different spotbeams. This scenario will also consider direct connections between user earth stations via satellite.

A significant body of work has been developed that addresses the behaviour of TCP over satellite links. However, little work has been conducted to date on IP multicast or reliable multicast, and how this is affected by the conditions which exist on satellite links. The current paper is a contribution to this area. The structure of the paper is as follows: Section 2 is an overview of the issues involved in the design of reliable multicast protocols and Section 3 provides a review of error correction techniques. The experimental Internet protocol PGM is introduced and described in Section 4. Sections 5 and 6 describe the theoretical models used in this paper: these are respectively a model of IP datagram loss over a satellite ATM link, and a model of the error performance of the reliable multicast protocol.

Finally Section 7 presents simulation results obtained using the event driven network simulation tool OPNET, and compares some of these results with results from the theoretical model of Sections 5 and 6.

2. Reliable multicast protocols

Reliable multicast protocols are of interest because they ensure that all recipients successfully receive data multicast from a source; in general they also ensure ordered and non-duplicated delivery of packets to the application layer. Since they provide an end-to-end service they are conventionally regarded as transport layer protocols in the context of the OSI Reference Model.

A wide range of reliable multicast protocols has been developed and described in the literature. One reason for this is that efficient multicast is a much more complex problem than efficient unicast, and consequently many multicast protocols have been developed for specific classes of application. Two examples of different application classes are delay-sensitive real-time applications and multicast file transfer, each of which has its own specific multicast requirements. A taxonomy of multicast protocols is described in [7], where they are referred to as "multicast transport protocols". This work has been taken forward in the context of satellite networks by Koyabe [4].

Following the structure used by Koyabe, some of the key issues that have to be considered by multicast protocols include:

- Data propagation: such as (a) whether the propagation is one-to-many, many-to-one, or many-to-many; (b) whether data transfers are one-way (outbound only) or two-way (return path required).
- Scalability: the number of data recipients (for example, of the order of a few tens or a few hundred thousands).
- Reliability: whether guaranteed delivery is required (for example, file transfer) or a certain loss rate can be tolerated (e.g. video streaming).
- Flow and congestion control: a multicast protocol needs to be sensitive to the differing needs of all receivers. This is particularly exacerbated if some recipients are connected via terrestrial links and some are connected by geostationary satellite links, because of the wide range of round trip times.

In this paper we focus on one of the above aspects, namely the protocol reliability and corresponding error correction capabilities.

3. Error correction techniques

There are two principal error correction approaches: forward error correction (FEC) and automatic retransmission request (ARQ). In the case of FEC, an original message is transmitted together with some redundant (parity) information to form a codeword, so that if part of the codeword is lost or corrupted the receiver can both detect and correct the error. The original message can thus be reconstructed from the redundant information in the codeword, provided that the number of errors is below a certain level. FEC is not by itself able to guarantee delivery of data, and has the further disadvantages of a coding overhead that is not needed when the channel error rate is low, reduced effective bandwidth of the channel and possibly an encoding or decoding delay. ARQ on the other hand is only able to detect errors in the original data, and if such errors occur the receiver requests a further copy of the data from the transmitter. ARQ has the advantage that it can guarantee data delivery, but can also suffer from significant delays when data has to be retransmitted, particularly over satellite links.

ARQ can be divided into two categories: these are Idle RQ and Continuous RQ. The latter can employ either a Selective Repeat or a Go-Back-N strategy. In the case of Selective Repeat, if a packet of data is lost or errored, the receiver requests and is sent a copy of only the errored packet. By contrast, with Go-Back-N, the receiver requests the transmitter to retransmit packets starting from the errored packet. The Selective Repeat strategy uses less network resources to transmit the data, but the Go-Back-N strategy can employ a simpler receiver architecture. We consider a Selective Repeat protocol as part of our analysis later in this paper.

A combination of FEC and ARQ can also be used, and this is called hybrid ARQ. In a Type-I hybrid ARQ scheme, parity data is transmitted with the original message so that errors can be both detected and corrected. If the number of errors is too high, so that they cannot be corrected, the receiver requests retransmission of the same codeword. In a Type-II hybrid ARQ scheme, if the receiver is unable to correct the errors in the received codeword then it requests transmission of further parity data until it has received sufficient to allow it to decode the original codeword. We consider a Type II hybrid ARQ scheme as a second error correction mechanism later in this paper.

We also consider proactive transmission of repair data, and here refer to this as a Type II hybrid ARQ scheme with proactive FEC. This technique has been considered in the context of reliable multicast by other authors [5, 6, 9].

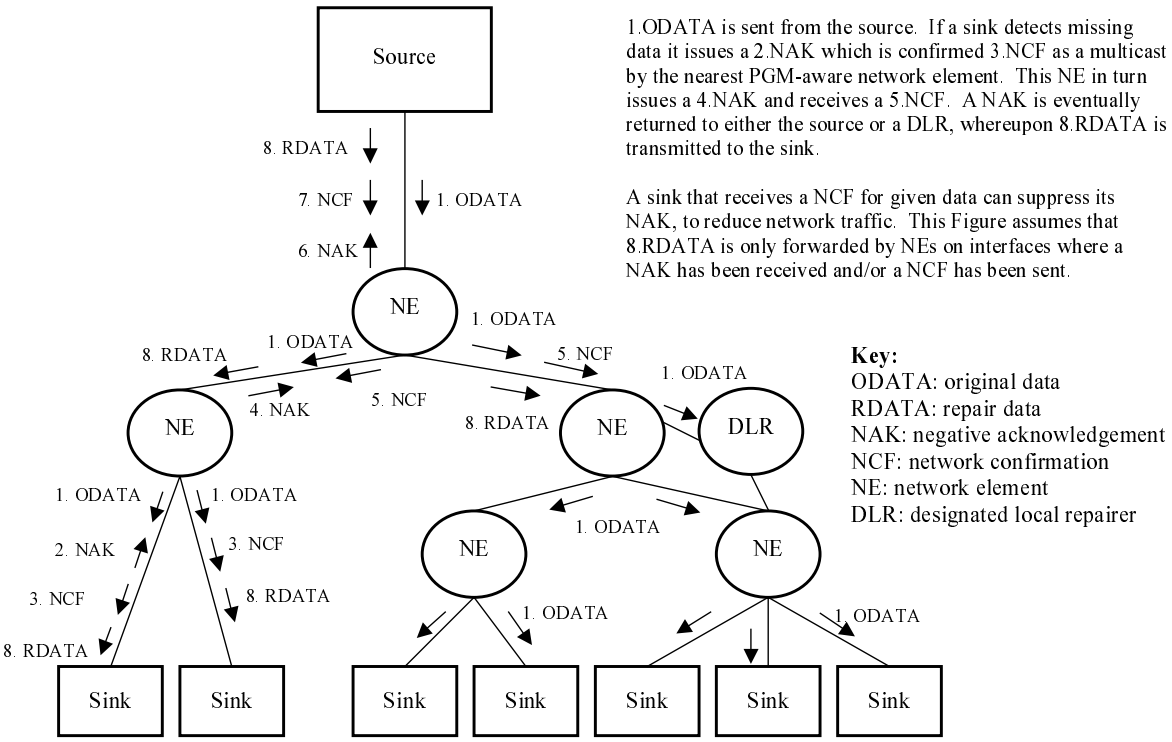

Figure 1: PGM: reliable multicast protocol

4. PGM

This Section outlines the reliable multicast protocol that has been simulated as part of the work described in this paper. The protocol is based on the experimental Internet protocol, PGM [10], and is used here to provide guaranteed ordered duplicate-free delivery of data from a single source to multiple sinks. Figure 1 illustrates the operation of PGM as defined by the RFC. This shows a data source and multiple sinks, together with network elements (NEs) that are "PGM-aware" and devices called Designated Local Repairers (DLRs). These latter devices maintain a copy of data transmitted by the source so that they can respond to requests from their nearby subtree for retransmission of lost data.

The normal flow of data is from a single source to multiple sinks, and is carried as *original data*. If a sink detects missing data, it issues multiple negative acknowledgements until it receives a Network Confirmation (NCF) from a network element. The purpose of the NCF is to notify the sink that the network element has taken responsibility for returning the NAK to the source. Consequently the network element in turn issues a NAK back "upstream" toward the source, and receives a NCF from the next NE. NAK suppression procedures are included in the protocol to avoid a NAK implosion when a large number of sinks fail to receive any given data. When the NAK arrives either at the source or a DLR, *repair data* is transmitted. This is multicast to receivers, except that the PGM NEs use the pattern of received NAKs to avoid sending repair data to subtrees where all sinks have received the original data.

Source path messages (SPMs) are also transmitted by the source: their function is to enable each NE to identify how to reach upstream NEs so that NAKs can be returned to the source or to DLRs.

The system modelled in the work described in this paper is significantly simpler than the full specification, and is shown in Figure 2. DLRs, NEs, and SPM and NCF messages have been omitted to simplify the model. These simplifications do not affect the fundamental principles of performance that this paper considers.

In our model, each sink maintains a timer for each negative acknowledgement so that the NAK can be re-issued if either the NAK or the repair data is lost. The source also maintains a separate timer so that if two or more NAKs for the same repair data are received within the timer period the repair data is not repeated.

In accordance with the PGM specification, a sliding window is employed to provide flow control: the source increments the window once no NAKs have been received for a period (defined by a third timer) for original data within the window increment. PGM provides two illustrative strategies for advancing the window, called "advance with data" and "advance with time". The former is intended for non-realtime applications that require complete delivery of data, and is consequently implemented here for our file application.

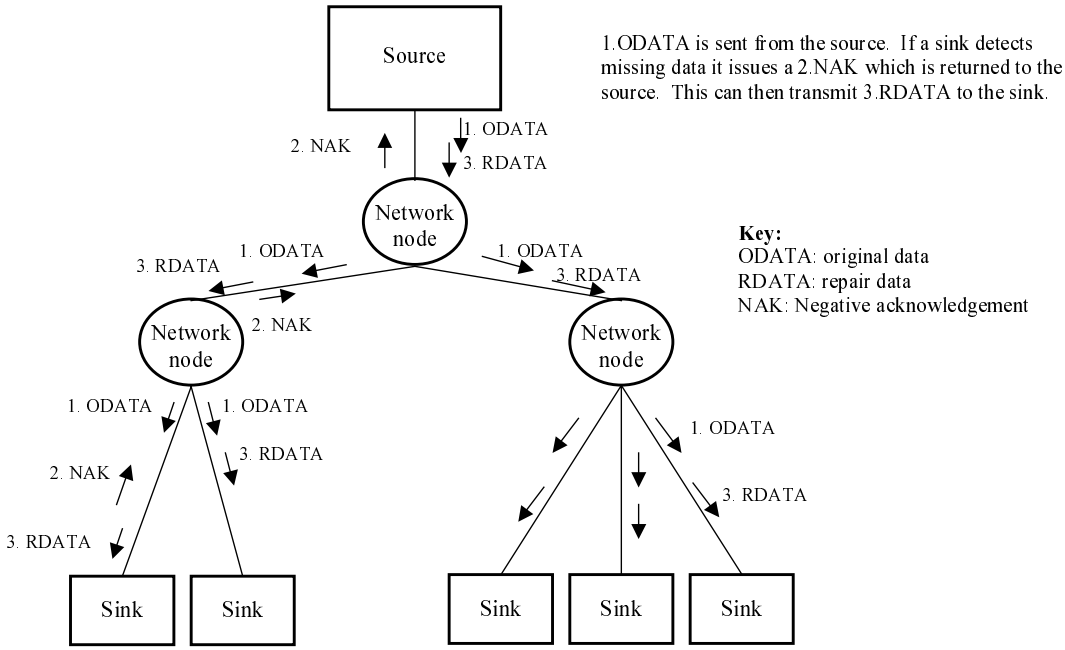

Figure 2: Protocol implemented in this work

The latter is intended for real-time streaming applications and is based on timely delivery of data even though this may involve loss of some packets (caused by the trailing edge of the sliding window moving forward even though some sinks may not received data that is older than the trailing edge).

Finally, a token bucket traffic shaper is also included (Figure 3), as required by PGM, to limit the bandwidth used. The traffic shaper implemented for most of the results presented here has a dual queue, with original data sent to the low-priority queue and repair data sent to the high priority queue. Packets in the high priority queue are always transmitted first.

Three error correction mechanisms have been considered:

- A Continuous RQ Selective Repeat protocol, referred to (following the PGM specification) as Selective NAK mode;
- A Type II hybrid ARQ protocol, referred to (as in the PGM specification) as Parity NAK mode;

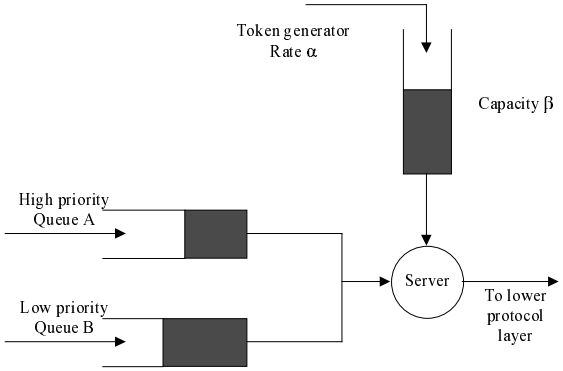

Figure 3: Dual queue traffic shaper

- A Type II hybrid ARQ protocol with proactive transmission of repair data, referred to here as Parity NAK mode with proactive FEC.

In a Type II hybrid ARQ protocol, data is transmitted with sufficient error control to detect a missing packet (in this case using sequence numbers). If the receiver detects an erasure then parity data is sent as repair data. The mechanism by which this parity data is generated in described by Rizzo [8].

The satellite system is shown in Figure 4. In the simulation, data rates of 2Mbit/s were used on the links, with a 130ms satellite one-way delay and 50ms of terrestrial link delay. The traffic shaper limited the traffic to about 20% of the network bandwidth.

5. Satellite ATM link model

Initially we analyse the link performance by considering the lower layers, from the satellite modem up to the IP layer (Figure 4).

A typical satellite communications link employs a convolutional encoder with Viterbi decoding. This reduces the link's effective bit error rate, but the nature of the Viterbi decoding means that this reduction in the error rate is at the cost of residual errors that occur in bursts. These error bursts either cause loss of ATM cells (if the errored bits occur in the ATM cell header) or payload errors (if the errored bits occur in the ATM cell payload). ATM cells with payload errors in turn cause the ATM Adaptation Layer to discard the entire IP datagram of which the errored cell was part. The probabilities of loss of an ATM cell and of an error in the cell payload can be calculated [2]. An IP datagram is lost if a cell loss or cell error occurs in any ATM cell that is carrying part of the datagram, or if the final cell of the preceding

AAL5 PDU is lost, and so the probability of loss of an IP datagram is given by:

$$P_{IPloss} = 1-(1-424\frac{p}{b})^{N}(1-(37+b)\frac{p}{b}) \qquad (1)$$

where p is the link bit error rate, b is the mean burst length, and N is the number of ATM cells required to carry the IP datagram. At low bit error rates this simplifies to:

$$P_{IPloss} \approx 424N\frac{p}{b} \qquad (2)$$

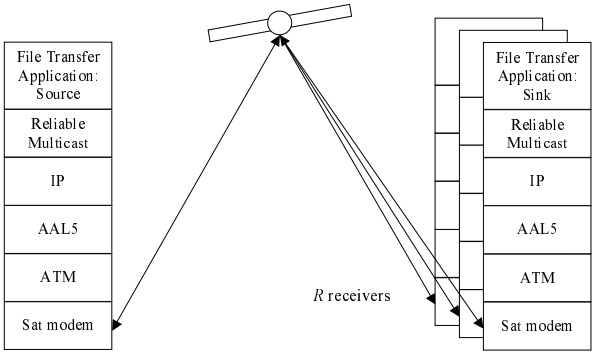

Figure 4: Satellite and protocol model

6. Theoretical analysis of protocol

The performance of a file transfer application over a reliable multicast protocol is analysed by considering the upper two layers of the protocol model of Figure 4.

Two key parameters of interest are the total network traffic used to transfer the data, and the number of negative acknowledgements (NAKs) received by the source. We use traffic volume (bytes) as a measure of network resource in the forward direction because, as we show later, the choice of packet size can have a significant impact on the total volume of data sent over the network from the source to the sinks. Conversely, NAKs are generally short packets, each probably only occupying one or two ATM cells, and so the number of these packets (rather than the number of bytes of network traffic) is an appropriate metric for the reverse link resources used.

Consider a file of size X bytes which is to be transferred using the satellite link. Let the file be transmitted as a set of packets of equal size, the final packet being padded out if necessary. If each application packet has a total header of H bytes from the multicast, IP and AAL5 layers combined, and is carried in N ATM cells then the number of application packets (or IP datagrams) used to transmit the file is $Q_{IP} = \left\lceil \frac{X}{48N-H} \right\rceil$ where the braces $\lceil\ \rceil$ mean "the smallest integer greater than". We assume the source is multicasting to R receivers.

Selective NAK mode

In Selective NAK mode, the mean number of repair data packets sent in response to the first NAK from R receivers is $1-(1-P_{IPloss})^{R}$. Some receivers will not receive the repair data and will issue a second NAK: the number of repair data packets then sent is $1-(1-P_{IPloss}^{2})^{R}$. Extending this, the total number of bytes required to transmit the entire file on the forward uplink, including both the original data and multiple rounds of repair data packets is:

$$B_{F-SN} = 53NQ_{IP}\left(1+\sum_{i=1}^{\infty}\left(1-(1-P_{IPloss}^{i})^{R}\right)\right) \qquad (3)$$

and the total number of NAKs on the reverse downlink is:

$$\begin{aligned} T_{R-SN} &= Q_{IP}R(P_{IPloss}+P_{IPloss}^{2}+..) \\ &= Q_{IP}\frac{RP_{IPloss}}{(1-P_{IPloss})} \end{aligned} \qquad (4)$$

Parity NAK mode

In Parity NAK mode the Q_{IP} packets are divided into transmission groups of k packets each, so the file transfer comprises $\left\lceil \frac{Q_{IP}}{k} \right\rceil$ transmission groups. If the sinks only request a single round of repair data W_1 then the total forward uplink traffic is given by:

$$B_{F-PN} = 53N\frac{Q_{IP}}{k}(k+W_1) \qquad (5)$$

and the number of NAKs on the reverse downlink is:

$$T_{R-PN} = \frac{Q_{IP}}{k}R\left(1-(1-P_{IPloss})^{k}\right) \qquad (6)$$

By considering the expected maximum value of the number of packets requested by any sink it can be shown that $W_1 = \sum_{i=1}^{k} i\left(F_R(i)-F_R(i-1)\right)$ where $F_R(l) = \left(F_1(l)\right)^{R}$ and $F_1(l) = \sum_{j=0}^{l} P_1(j)$ is the CDF of the probability of a single receiver losing j packets $P_1(j) = \binom{k}{j} P_{IPloss}^{j}(1-P_{IPloss})^{k-j}$.

Equation (5) can also be extended to cases of high bit error rate or large numbers of receivers where there is a significant probability of receivers issuing second or further NAKs.

Parity NAK mode with proactive FEC

Here, s proactive FEC packets are transmitted at the same time as the k packets of original data in each transmission group. In general some packets will not be received by each sink, but a NAK only needs to be issued by a receiver when the number of lost packets is greater than s. Consequently, for the complete file transfer the total number of NAKs on the reverse downlink in this mode is, assuming only a single round of repair data is required by the receivers:

$$T_{R-PNPF} = \frac{Q_{IP}}{k} R\left(1 - \sum_{i=0}^{s} P(i)\right) \qquad (7)$$

where $P(i) = \binom{k+s}{i} P_{IPloss}^{i} (1 - P_{IPloss})^{k+s-i}$. The ratio $\frac{k+s}{k}$ is defined by Rubenstein [9] as the proactivity factor.

7. Comparison of theory & simulation

The simulation results are now compared with the theoretical curves for the forward uplink and reverse downlink traffic. In addition, two aspects of the protocol design have been investigated using the simulation:

- The impact of the token bucket scheme used to provide traffic management;
- The impact of the decoding delay on the packet transfer delay when using Parity NAK mode.

The forward uplink traffic is shown in Figure 5 as a function of the size of each application packet, represented by N the number of ATM cells required to carry the encapsulated application packet. For any given BER, the forward traffic in Parity NAK mode is less than in Selective NAK mode. This can be illustrated by two sinks each of which loses one packet within a transmission group, but they are different packets. Parity NAK can recover this loss by transmitting one repair data packet, but Selective NAK needs to resend each of the two packets as repair data, and therefore generates more traffic.

We note that as the BER increases, the range of N that gives acceptable performance gets narrower; in other words, the worse the link conditions, the more important it is to transmit the file in packets that are correctly sized: if the packets are too small then the protocol overhead limits the network performance, whereas if the packets are too large the probability of packet loss and therefore retransmission becomes too high. We also see that the range of N that gives acceptable performance (i.e. close to the minimum for any given BER) is wider for Parity NAK mode than for Selective NAK mode.

The reverse downlink traffic is shown in Figure 6. In Selective NAK mode the NAK volume is essentially independent of N because for reasonably low values of bit error rate p and ignoring the protocol overhead ($48N >> H$) equation (4) reduces to an expression that is independent of N: $T_{R-SN} \approx \frac{424}{48} RX \frac{p}{b}$. The middle graph shows that Parity NAK is not worse than Selective NAK, and is better for some values of BER and N. The bottom graph shows for Parity NAK mode with Proactive FEC that by appropriate selection of proactivity factor ρ the NAK volume can be reduced to an arbitrarily low value.

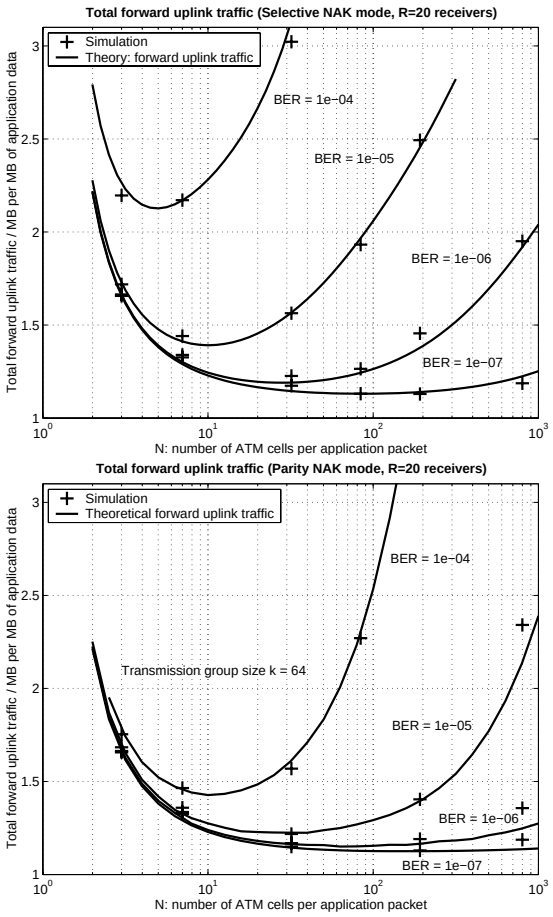

Figure 5: Total forward uplink traffic – Selective NAK mode (top) and Parity NAK mode (bottom)

We now consider the design of the token bucket scheme. For illustration, results are presented from three schemes:

- The dual queue token bucket (Figure 3), with repair data sent to the high priority queue and original data sent to the low priority queue, so repair data is consequently always given priority over original data.
- A single queue token bucket, with both original data and repair data sent to the same queue with the same priority.

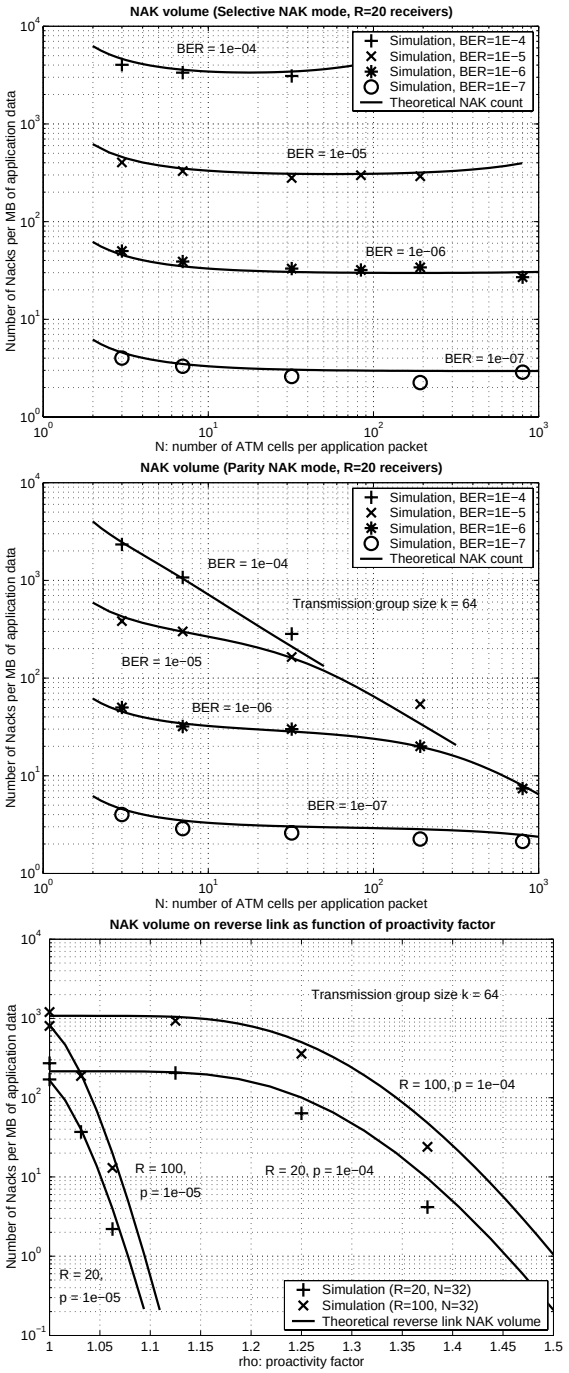

Figure 6: Reverse downlink traffic – Selective NAK mode (top), Parity NAK mode (middle), and Parity NAK with proactive FEC (bottom)

- No token bucket. Although this is not permitted by the PGM specification it has been modelled for comparison purposes.

Figure 7 shows the file transfer time for Selective and Parity NAK modes for each of these three token bucket schemes. Selective NAK has the worst performance because of its high volume of repair data (as seen in Figure 5). Of the three options, the lowest file transfer time is provided, unsurprisingly, by the null token bucket; however this is only achieved at the cost of allowing the protocol to use more of the available link bandwidth than would be allowed by a traffic shaper. The dual queue bucket has better performance than the single queue bucket because it allows repair data to be prioritised and therefore sent back to the sinks more rapidly. Indeed, the performance of the single queue bucket with Selective NAK is so poor that the sink NAK timers expire and request retransmission of repair data, overloading the link still further. If the sinks' NAK timer value is increased to compensate for this effect the performance can be improved: compared to the dual queue bucket, the overall file transfer time is still higher, although the delay per packet is reduced. These results occur because the overall mean traffic on the forward link is reduced.

These effects are less pronounced with Parity NAK mode, because the volume of repair data is significantly less at high bit error rates.

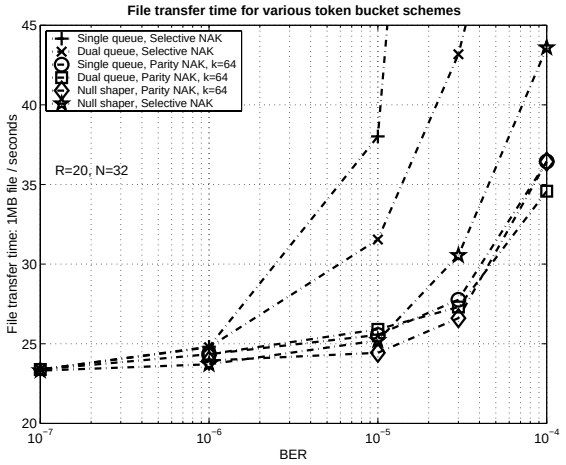

Figure 7: File transfer time for various token bucket schemes

The mean packet delay is of interest in data streaming applications, although it is not an important metric for a file transfer application. Such a streaming application may use PGM's "advance with time" window advance strategy, although this can result in loss of data. However, there may be applications that require reliable delivery of data but also require *reasonable* timeliness. In this case they may use "advance with data", and the results for delay now presented in Figure 8 are of interest. As with the file transfer time, we see that the null shaper clearly results in the lowest packet delay, but this is at the expense of link bandwidth used. For Selective NAK the dual queue bucket again has better performance than the single queue bucket since it prioritises repair data. For Parity NAK mode there is virtually no difference between the single and dual queue buckets.

At medium to high BERs, the advantage of Selective NAK mode in that it can immediately request repair packets is more than cancelled out by the fact that it requires more repair data to be sent. Parity NAK on the other hand cannot request repair data until the end

of the transmission group, so suffers a greater delay; however, the repair data volume is less for Parity NAK mode than for Selective NAK. We therefore see that in this protocol where the network bandwidth is limited by the token bucket Parity NAK has better packet delay performance than Selective NAK mode.

Finally, we consider the effect of a delay in decoding packets when erasure codes are used. Decoding times of the order of a few milliseconds have been reported in tests [8]. Here we have implemented a delay for each packet within a transmission group that has to be decoded to recover the original data. Figure 9 shows how the mean packet delay varies with this decoding delay. Except at very high BERs (10^{-4}) the additional packet delay is negligible, illustrating that erasure codes are suitable for delivery of packets in or close to real time.

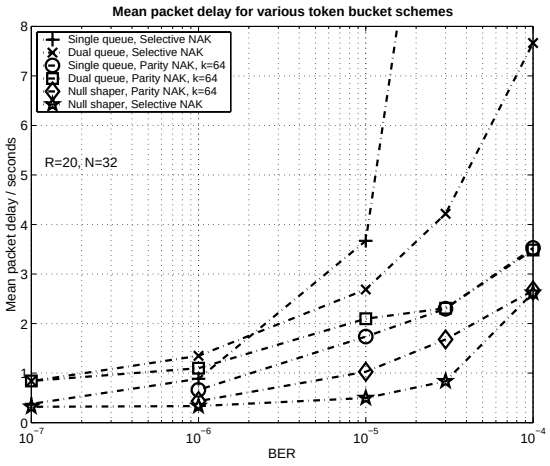

Figure 8: Mean packet delay for various token bucket schemes

8. Conclusions

When multicast is used to provide reliable services to potentially hundreds of thousands of recipients, transmitted data will be incorrectly received by a significant number of recipients, and the error recovery mechanism becomes a critical aspect of a reliable multicast protocol. This paper shows that Parity NAK mode has lower forward traffic volume than Selective NAK mode, and is less sensitive to the best selection of packet size at any given BER. Parity NAK also results in a lower file transfer time. Proactive FEC reduces the reverse link negative acknowledgement traffic. Finally, a dual queue token bucket provides superior performance to a single queue bucket.

Acknowledgements

This work was supported by the UK Engineering and Physical Sciences Research Council, and by the EU Information Society Technologies GEOCAST project, IST 11754.

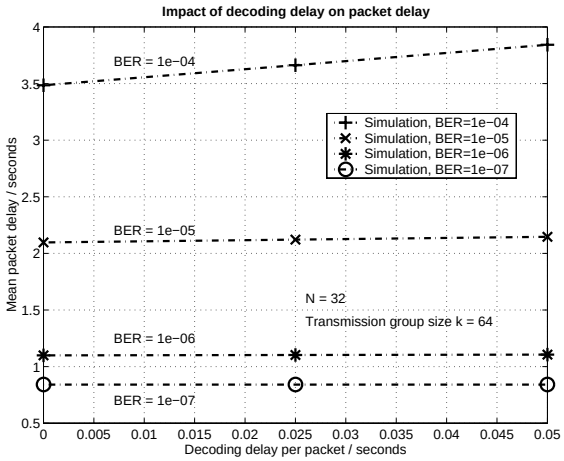

Figure 9: Mean packet delay as a function of decoding delay

References

[1] I.F. Akyildiz and S.H. Jeong, "Satellite ATM networks: a survey," *IEEE Communications Magazine*, July 1997, pp.30-43.

[2] M.P. Howarth, H. Cruickshank and Z. Sun, "Unicast and multicast IP error performance over an ATM satellite link," *IEEE Communications Letters*, Vol. 5, No. 8, August 2001, pp.340-342.

[3] S. Iyengar, H. Cruickshank and Z. Sun, "Security issues in IP multicast over GEO satellites," *19th AIAA Int Comms Satellite Systems Conf and Exhibit*, 17-20 April 2001, Toulouse, France.

[4] M. Koyabe and G. Fairhurst, "Reliable multicast via satellite: a comparison survey and taxonomy," *International Journal of Satellite Communications*, Vol. 19, No. 1, Jan 2001, pp.3-28.

[5] D. Li and D.R. Cheriton, "Evaluating the utility of FEC with reliable multicast," *Proc. 7th Intl Conf on Network Protocols*, Nov 1999, pp.97-105.

[6] J. Nonnenmacher, E.W. Biersack and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Trans. Networking*, Vol. 6, No. 4, Aug 1998, pp.349-361.

[7] L. Obraczka, "Multicast transport protocols: a comparison survey and taxonomy," *IEEE Communications Magazine*, Jan 1998, pp.94-102.

[8] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM Computer Communication Review*, April 1997, pp.24-36.

[9] D. Rubenstein, J.Kurose and D. Towsley, "A study of proactive hybrid FEC/ARQ and scalable feedback techniques for reliable, real-time multicast," *Computer Communications*, Vol. 24, 2001, pp.563-574.

[10] T. Speakman et al., "PGM reliable transport protocol specification," IETF RFC 3208, Dec 2001.

[11] F. Yegenoglu, R. Alexander and D. Gokhale, "An IP transport and routing architecture for next-generation satellite networks," *IEEE Network*, Sep 2000, pp.32-38.