Quantitative Analysis of Restricted Reliability Protocols for Constant Rate Traffic

Michael P. Howarth and Zhili Sun, *Member, IEEE*

***Abstract*—Restricted reliability, or limited reliability, transport layer protocols reduce but do not eliminate the packet loss probability seen by higher layers. Forward error correction (FEC) is employed to control the packet loss rate, but automatic repeat request (ARQ) is not used, either because no reverse channel exists or because it does not support the required quality of service.**

This letter analyzes the effect of FEC on packet loss probability and protocol delay and jitter. We demonstrate the tradeoff between parameters, showing how increased network traffic can be traded against reduced delay and jitter. The theoretical results are confirmed by simulation.

***Index Terms*—Data streaming applications, FEC, jitter, multicast, packet loss rate, unicast.**

I. Introduction

RESTRICTED reliability, or limited reliability, protocols are ones in which forward error correction (FEC) data is sent proactively together with original data; the FEC data can be used to repair most but not all packet losses. This technique thus gives an improvement over a protocol with no FEC, but is not fully reliable since there is no automatic repeat request (ARQ).

The motivation for studying and using restricted reliability protocols is that they provide an improvement in the error rate observed by the application but do not require a reverse channel. This makes them appropriate in situations where either no such channel exists, or a channel exists but has low bandwidth (e.g., satellites). They are also suitable for real-time applications such as video or audio streaming, where reliable data transfer using ARQ is not feasible because of the time to request and retransmit lost or corrupted data [1]. Restricted reliability protocols can be used for either unicast or multicast; a further advantage in the case of multicast restricted reliability protocols [2] is that they avoid multicast NAK implosion. Biersack [3] showed that FEC is beneficial in a heterogeneous traffic mix: the increased ability to recover lost packets more than cancels any increased network loss rate caused by the FEC packets.

II. Analysis

The protocol takes a block of k data packets and encodes s FEC packets using an erasure correcting code (e.g., [4], [5]).

Manuscript received February 24, 2003. The associate editor coordinating the review of this letter and approving it for publication was Dr. J. Kim. This work was supported by the EU Information Society Technologies program under the GEOCAST project IST-1999-11754 and the ICEBERGS project IST-2000-31110.

The authors are with the Centre for Communication Systems Research, University of Surrey, Guildford GU2 7XH, U.K. (e-mail: m.howarth@surrey.ac.uk; z.sun@surrey.ac.uk).

Digital Object Identifier 10.1109/LCOMM.2003.817336

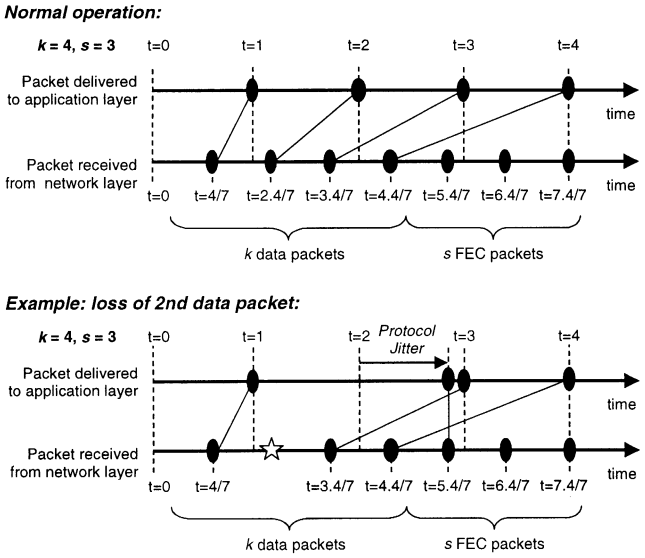

Fig. 1. Protocol operation: receipt and delivery of packets at receiver.

The set of $k+s$ packets is then transmitted as a systematic code consisting of the k data packets followed by the s FEC packets. The network has a packet loss probability p and so in general some packets are lost. At the receiver, all k data packets can be recovered provided that at least k of the set of $k+s$ are received. If more than s packets are lost, then none of the lost data packets can be recovered.

A. Assumptions

We assume data packets are generated by the application at a constant rate. We further assume that the $k+s$ packets are also transmitted at a constant rate, to avoid network traffic bursts. To normalize the models developed here, we assume that the k data packets represent streaming application data from k units of time, irrespective of the number s of FEC packets. The network packet transmission rate is thus $(k+s)/k$ packets per unit time (Fig. 1).

We assume the application requires ordered delivery of data packets. We also ignore any application buffering or interleaving, since we here consider only the effect of the transport layer protocol. We assume packet losses are independent. We ignore coding delay, since this can be performed rapidly with modern processors. We also ignore network delay and network jitter, to focus on the effect of proactive FEC on the protocol.

We now derive expressions for application packet loss probability, protocol delay and protocol jitter.

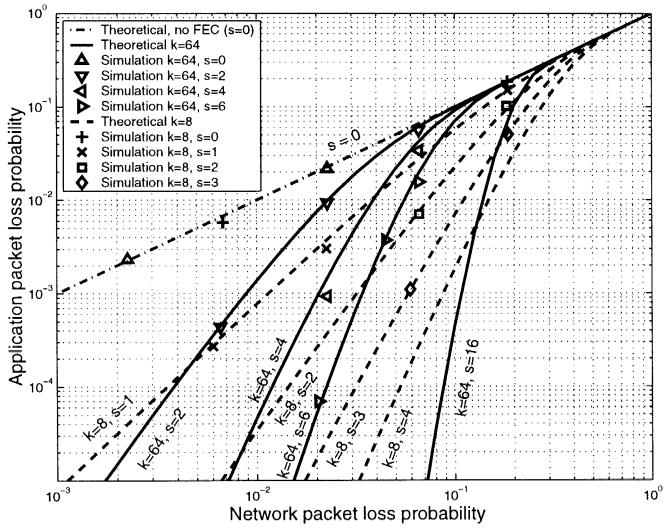

Fig. 2. Variation of application packet loss probability with network packet loss probability.

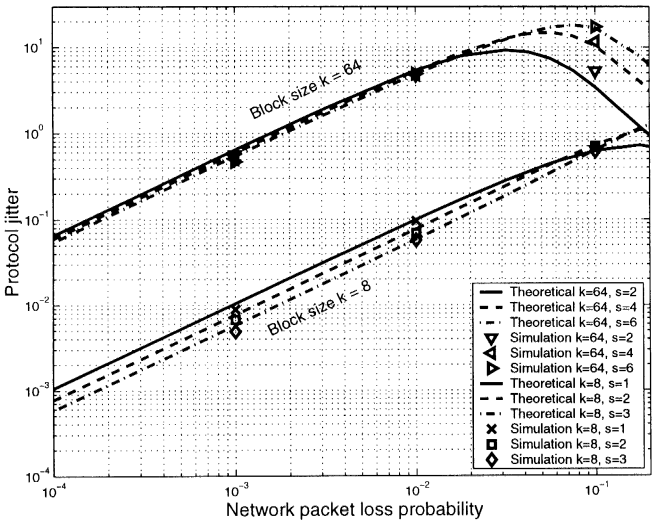

Fig. 3. Protocol jitter as function of network packet loss probability.

B. Application Packet Loss Probability

Following the analysis of [6] for a FEC layer, a data packet or application packet is lost if (a) it is not received at the receiver and (b) more than $s-1$ of the other $k+s-1$ packets in the set are also lost so that the erasure correcting code is unable to recover the packet. The application packet loss probability P_A is, therefore, given by:

$$P_A = p\left(1 - \sum_{j=0}^{s-1} \binom{k+s-1}{j} p^j (1-p)^{k+s-j-1}\right). \quad (1)$$

This function is plotted in Fig. 2. The expression has also been validated by simulation using the event driven simulation tool Opnet, and these simulation results are shown in the figure.

C. Protocol Delay

We define protocol delay Δ_{mean} to be the mean packet delay, due solely to the restricted reliability protocol, in delivering a packet to the receiving application, assuming zero packet loss. Ignoring network delays, data packet j is sent at time $jk/(k+s)$ and delivered to the application at time j (Fig. 1). The delay for this packet is therefore $j(1-k/(k+s)) = js/(k+s)$ and so Δ_{mean} is

$$\Delta_{\text{mean}} = \frac{1}{k}\sum_{j=1}^{k} j\frac{s}{k+s} = \frac{s(k+1)}{2(k+s)}. \quad (2)$$

The maximum delay under these conditions is for the final data packet of the block $\Delta_{\max} = ks/(k+s)$.

D. Protocol Jitter

Protocol jitter, due solely to the effect of the restricted reliability protocol, occurs when one or more of the k data packets are lost, and repair therefore has to await the arrival of one or more FEC packets (Fig. 1). We define the mean jitter $\overline{J}$ as the mean difference between the time a packet is due to be delivered to the application and the time it is actually delivered to the application:

$$\overline{J} = \frac{1}{x}\sum_{\text{packets}} (t_{\text{actual}} - t_{\text{due}})$$

where x is the number of data packets in the block that are actually delivered, and $t_{\text{due}} = j$ for data packet j. We note that, taking into account only the effect of the restricted reliability protocol, $\overline{J}$ is always positive [7]. $\overline{J}$ is derived in the Appendix as the sum of two terms, $\overline{J}_A$ and $\overline{J}_B$.

$\overline{J}$ is plotted in Fig. 3 for $k = 8$ and $k = 64$. Again, the expression has been validated by simulation, as shown in the figure. The maximum occurs when the network packet loss is so high that the FEC is unable to recover the lost packets: as soon as more than s losses have been detected all received packets can be delivered to the application, resulting in decreased jitter. However, this is not a good operating point since the FEC is providing virtually no improvement in application packet loss rate.

III. Application

We illustrate the use of these equations with an example. Say we wish to use restricted reliability to gain an improvement of two orders of magnitude in the packet loss probability, e.g., network loss $\text{rate} = 10^{-2}$, application packet loss $\text{rate} = 10^{-4}$. From Fig. 2, this can be achieved with either $k = 8$ and $s = 2$, for a 25% network traffic overhead; or $k = 64$ and $s = 4$, for a 6.25% network traffic overhead.

To compare the protocol delay and protocol jitter of these two options, consider an audio protocol which transmits frame updates every 7.5 ms. The mean protocol delay is from (2), at zero BER:

- For $k = 8$, $s = 2$, mean $\text{delay} = 0.9$ application packets or 6.75 ms; or
- For $k = 64$, $s = 4$, mean $\text{delay} = 1.91$ application packets or 14.3 ms.

The protocol jitter is, from Fig. 3:

- For $k = 8$, $s = 2$, mean jitter $= 0.08$ application packets or 0.6 ms; or
- For $k = 64$, $s = 4$, mean jitter $= 5.3$ application packets or 39.8 ms.

The maximum jitter occurs when the first packet in a block is lost and repair has to wait until the final FEC packet is received, and is therefore $k - 1$ application packets; for the two options this is respectively 7 and 63 application packets (53 ms and 473 ms).

IV. Conclusions

Our novel contribution in this letter has been to derive and apply expressions for application packet loss probability, delay and jitter for restricted reliability protocols. We have shown how increasing the number of FEC packets transmitted with a block improves the application packet loss rate. We have illustrated how a small block size has a reduced protocol delay and jitter but requires a larger network traffic volume. A large block size on the other hand has smaller network traffic overhead but at the cost of larger delay and jitter. The quantitative analysis has been confirmed by simulation study.

Appendix
Derivation of Protocol Jitter

We approximate protocol jitter as the sum of two distinct components: (a) when the number of data packets lost, m, is less than or equal to the expected number of FEC packets received, $s(1-p)$, so that all the data packets can be recovered; (b) when $m > s(1-p)$, so that no lost data packets can be recovered, and only the $k-m$ data packets actually received can be delivered to the application.

A. $m \leq s(1-p)$

Let data packet j be the first lost packet in the block. If m data packets are lost, all lost packets can be recovered when FEC packet $\lfloor m/(1-p) \rfloor$ arrives at time $(k + \lfloor m/(1-p) \rfloor)\, k/(k+s)$. The jitter is zero for data packets 1 through $j-1$, and for any data packets that are due to be delivered to the application after the arrival of FEC packet $\lfloor m/(1-p) \rfloor$. The mean jitter averaged over the block of k data packets when the first lost packet is j is thus $\Phi = (1/k) \sum_{i=j}^{u} [(k + \lfloor m/(1-p) \rfloor)\,(k/(k+s)) - i]$ where $u = \lfloor (k + \lfloor m/(1-p) \rfloor)\, k/(k+s) \rfloor$. The probability of losing m out of k data packets with the first lost packet at j is $\binom{k-j}{m-1}(1-p)^{k-m}p^m$ for $1 \leq j \leq k-m+1$. Thus the mean jitter when $m \leq s(1-p)$ is

$$\overline{J}_A = \sum_{m=1}^{\lfloor s(1-p) \rfloor} (1-p)^{k-m} p^m \sum_{j=1}^{k-m+1} \binom{k-j}{m-1} \Phi. \qquad (3)$$

B. $m > s(1-p)$

To simplify the analysis assume the m lost data packets are evenly spaced between packets 1 and k. The first lost packet is therefore approximately $j = round\,(k/(m+1))$. As soon as the receiver determines that $> s(1-p)$ packets have been lost and therefore it cannot repair any lost data, it delivers all packets. The mean jitter averaged over the $k-m$ received data packets is thus $\Psi = (1/(k-m)) \sum_{i=(k/(m+1))}^{v} [((v+1)k/(k+s)) - i]$ where $v = \lfloor (k/(m+1))\,(1 + s(1-p)) \rfloor$. The probability of losing m out of k data packets is $\binom{k}{m}(1-p)^{k-m}p^m$, and so the mean jitter when $m > s(1-p)$ is

$$\overline{J}_B = \sum_{m=\lceil s(1-p) \rceil}^{k} \binom{k}{m} (1-p)^{k-m} p^m \Psi. \qquad (4)$$

References

[1] C. Perkins *et al.*, "RTP payload for redundant audio data,", IETF RFC2198, Sept. 1997.

[2] H. Linder and H. D. Clausen, "Scalable multicast data distribution for different transport service classes," in *Proc. IEEE Int. Perform. Comput. & Commun. Conf.*, 1998, pp. 435–441.

[3] E. W. Biersack, "Performance evaluation of forward error correction in an ATM environment," *IEEE J. Selected Areas Commun.*, vol. 11, pp. 631–640, May 1993.

[4] A. J. McAuley, "Reliable broadband communication using a burst erasure correcting code," in *Proc. ACM SIGCOMM*, Philadelphia, PA, Sept. 1990, pp. 297–306.

[5] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM Comput. Commun. Rev.*, pp. 24–36, Apr. 1997.

[6] J. Nonnenmacher, E. W. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Trans. Networking*, vol. 6, pp. 349–361, Aug. 1998.

[7] C. J. Hamann, "On the quantitative specification of jitter constrained periodic streams," in *Proc. 5th IEEE MASCOTS*, 1997, pp. 171–176.