Interleaved FEC/ARQ coding for QoS multicast over the Internet

Codage FEC/ARQ interlacé pour de la multidiffusion de qualité sur l'Internet

Donghui Chen, Bo Rong, N. Shayan, M. Bennani, J. Cabral, M. Kadoch, and A.K. Elhakeem*

The technology of real-time reliable multicast over a best-effort service network has become more popular recently. In this paper, a new technique is introduced that integrates word interleaving, forward error correction (FEC) and automatic repeat request (ARQ) to mitigate the error and loss effects encountered in wired and wireless Internet applications. For multicast video sessions spanning tens of routers, the integration of interleaving, FEC and ARQ as well as fine tuning of the various parameters of each mechanism will play a major role in the move towards successful deployment. In this work, the basic algorithms for integrated error control are first derived. Then the corresponding real-time implementations for an experimental test bed to achieve the practical performance of these techniques are completed. The performance of the scheme is analyzed, and laboratory results are compared with analytical results.

La technologie de la multidiffusion en temps réel et fiable sur un réseau de service meilleur effort est récemment devenue plus populaire. Dans cet article, une nouvelle technique qui intègre l'interlacement de mots, la correction d'erreur directe (FEC) et la demande automatique de répétition (ARQ) afin de limiter les effets d'erreur et de perte rencontrés dans des applications Internet avec et sans fil est introduite. Pour des sessions multidiffusion de vidéo portant sur des dizaines de routeurs, l'intégration de l'interlacement, de FEC et de ARQ ainsi que le réglage de précision de tous les paramètres de chaque mécanisme jouera un rôle majeur dans un déploiement réussi. Dans ce travail, les algorithmes de base pour un contrôle d'erreur intégré sont d'abord dérivés. Puis, les implémentations correspondantes en temps réel pour un banc d'essai expérimental afin d'atteindre les performances pratiques de ces techniques sont effectuées. La performance du schème est analysée et des résultats en laboratoire sont comparés avec des résultats analytiques.

Keywords: ARQ, FEC, interleaving, multicast, Reed-Solomon codes

I. Introduction

Nowadays, multicast is becoming more and more important to real-time multimedia communication applications, such as interactive video conferencing, video-on-demand systems, and so on. These applications usually have stringent quality of service (QoS) requirements in terms of bandwidth, end-to-end delay, delay jitter and packet loss rate. As one of the most challenging aspects of multicast, the technology of controlling packet loss has attracted more and more attention [1]–[6]. Congestion, tunnelling, and lack of high-speed backbones are among the reasons for occasional data loss. On the wireless side, bursty channel errors combined with packet loss make stronger error and loss control more crucial. Numerous techniques have been suggested for error and loss control over the Internet. In the past, mere packet replication was usually suggested to combat the best-effort delivery and subsequent transportation-layer protocol data unit (TPDU) loss of the Internet, but now this approach has given way to powerful erasure codes. These erasure codes, such as Reed-Solomon (RS) codes, have strong inherent erasure-correction capability [7].

Although work has been done on employing forward error correction (FEC) and automatic repeat request (ARQ) in multicast [8]–[9], few actual buildups and experiments which integrate all the techniques, such as interleaving, FEC and ARQ, have been reported. This paper concentrates on an actual experiment involving interleaved RS/ARQ error correction. The results show that RS/ARQ combined with interleaving can provide strong multicast error control and excellent QoS guarantees at high user data rates.

The rest of the paper is organized as follows. First, we introduce our interleaved FEC/ARQ error correction scheme in Section II. Then the experimental results are demonstrated and analyzed in Section III. In Section IV, the theoretical analysis of our scheme is presented. Finally, Section V summarizes our results.

II. Interleaved RS/ARQ error correction scheme

Our scheme is based on client/server architecture, and the main entities of the new technique are the server and client algorithms which reside on top of the Internet user datagram protocol (UDP) layer. The server routine typically exists at the sender of a video/audio multicast session or at one of the intermediate routers (called the domain receiver (DR) in multicast terminology). The client routine exists within the receiver part of the DR or the end user.

The frames transmitted from the sender (first sender or intermediate DR) are shown in Fig. 1. Fig. 1 shows clearly how the application multicast data packets are encapsulated within the frames. Each block of data, consisting of 217 bytes, is given a sequence number ranging from 1 to 65 536. The sequence number field in a corresponding word is preceded by a type field. So far, we have only two designations, where 11111111 and 10101010 denote a fresh and a retransmitted word respectively. A 1-byte data length (DL) field is also needed to indicate the end of receiving because the data may be less than 217 bytes. If the data are less than 217 bytes, the remainder of the 217 bytes is filled with padding. A 2-byte cyclic redundancy check (CRC) field is also formed at the end of the word. This CRC operates over the whole

*Donghui Chen, N. Shayan, and A.K. Elhakeem are with the Department of Electrical and Computer Engineering, Concordia University, Montreal, Quebec H3G 1M8. Bo Rong, M. Bennani, J. Cabral, and M. Kadoch are with the Département de génie électrique, École de technologie supérieure, Université du Québec, Montréal, Québec H3C 1K3. E-mail: bo.rong@lagrit.etsmtl.ca

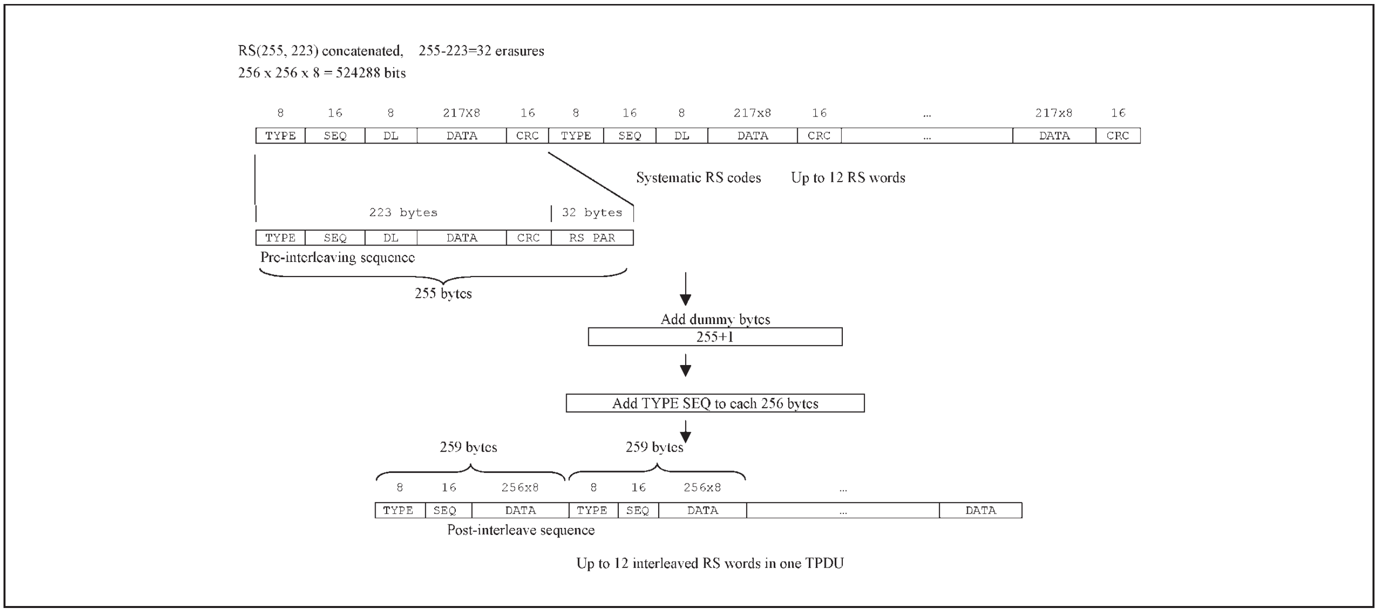

Figure 1: *Transportation-layer protocol data unit (TPDU) format.*

word. The resulting 223 bytes are fed to the $(255, 223)$ Reed-Solomon encoder routine [1], which outputs a corresponding 255 bytes that we call an encoded word. The RS code is assumed to be systematic, such that the 32 parity bytes are added by the encoder at the end of the word. One byte of dummy data is added to obtain a 256-byte encoded word.

Interleaving is next applied to a record of 256 such encoded RS words to yield a corresponding interleaved record. The first interleaved word of such a record consists of the first bytes of each of the 256 pre-interleaving words. The 256-th interleaved word is the concatenation of the 256-th bytes of all the pre-interleaving words. It is also possible to have a smaller interleaving depth such as 128, in which case the first interleaved word will encompass the first and second bytes of every pre-interleaving word. Then the first 128 bytes of the 128-th interleaved word are bytes numbered 255 in the original pre-interleaving words, and the second 128 bytes of the 128-th interleaved word are bytes numbered 256 in the original pre-interleaving words.

If the interleaving depth is 256, it is easy to understand that up to 32 consecutive interleaved words can be lost without any ARQ retransmission request from the receiver. Because of interleaving, if any 32 words are lost, only 32 of the 255 bytes of each corresponding pre-interleaving word will be lost. Because the RS decoder can conceal up to $255 - 223 = 32$ lost bytes, such loss is not detrimental and will not result in retransmission requests. For error concealment for multicast traffic over the Internet, minimizing the need for retransmissions is an essential asset, which subsequently leads to the minimizing of no acknowledgement (NAK) and repair-packet implosions [2]. Returning to Fig. 1, we see that the server routine groups up to 12 interleaved words into one UDP data unit (TPDU). A 2-byte interleaved sequence number field is added to each interleaved word, preceded by a 1-byte field denoting an interleaved word, i.e., 11111111. Although each original RS-encoded word carries a type field and a sequence field, these fields are scrambled later by the interleaving, and the need arises for new type and sequence fields. Occasionally or at the request of the client, retransmitted words are also included in some TPDUs. If an interleaving depth of 256 is used, each TPDU contains eight interleaved words; in this case four consecutive TPDUs can be lost without any retransmissions.

Fig. 2(a) shows the byte processing of a transmitted multicast file, as outlined above. Meanwhile, Fig. 2(b) underlines the processing of retransmission packets and shows how to combine them with new interleaved words. Most of the processing above is done offline by multicast applications. The whole video or voice file is FEC-encoded, interleaved and stored before the real-time multicast is started. The procedure of mixing repairs and interleaving words has to be executed in real time when the multicast session commences.

The format of the NAK frame used by the processing in Fig. 2(b) is shown in Fig. 3. As mentioned before, for the multicast applications at hand, this is the only packet type that the client can send. However, for a video conferencing system, data may also be sent from the client to the server. In Fig. 3, we use 01010101 as the value of the type field to denote NAK messages. A 1-byte field denoting the data length of the NAK message is also added. The subsequent fields give the sequence numbers of the lost words as requested by the client. A final 1-byte CRC field is used for error checking at the server. The client routine in Fig. 4(a) shows the process that takes place once a record of data is read from the UDP layer. Blocks of 259 bytes are sampled from such a record. If the type field indicates a repair word, and if further CRC indicates correct reception, the first 4 bytes are stripped and the following 217 bytes are sent to the receiving buffer. The DL field may be used to separate real data from padding. If the type field indicates an interleaved word, this interleaved word is sent to the de-interleaving buffer. A timer is reset upon the arrival of the first byte of a certain de-interleaved RS word (this may not be the first byte of the word).

Routinely, the de-interleaved RS words are checked one after another. Let E be a suitably selected number in the range 1 to $(N - K)$, where $N = 255$, $K = 223$. TH1 is a period of time corresponding to a number of bits in the range $((K+E)*256*8)$ to $(N*256*8)$, while TH2 corresponds to the range $(N*256*8)$ to $((N+E)*256*8)$. If, in a certain RS word, the first $K = 223$ bytes have been received before the expiry of a certain time out TH1 on the timer above, and CRC indicates correct reception, the first 4 bytes are stripped and the following 217 data bytes are delivered to the receiver buffer. If CRC checking fails, erasure-based RS decoding is tried, and later CRC checking is performed again. If CRC works in this second trial after RS decoding, the 217 bytes are delivered to the receiver buffer. If CRC fails for the second time in a row, the client waits for more bytes before trying RS and CRC decoding again.

If $(K + E)$ bytes out of $N = 255$ bytes are received (not necessarily the first K bytes) before the expiry of time out TH2, erasure-based RS decoding is performed first, after which CRC checking is verified and 217-byte data delivery to the receiver buffer takes place. If CRC checking fails in the latter case, or time out TH2 is exceeded without receiving $(K + E)$ bytes, then a NAK (repair request) is formulated and sent to the server. Because of interleaving, one has to pay for the loss of resilience by waiting a longer time before a meaningful number

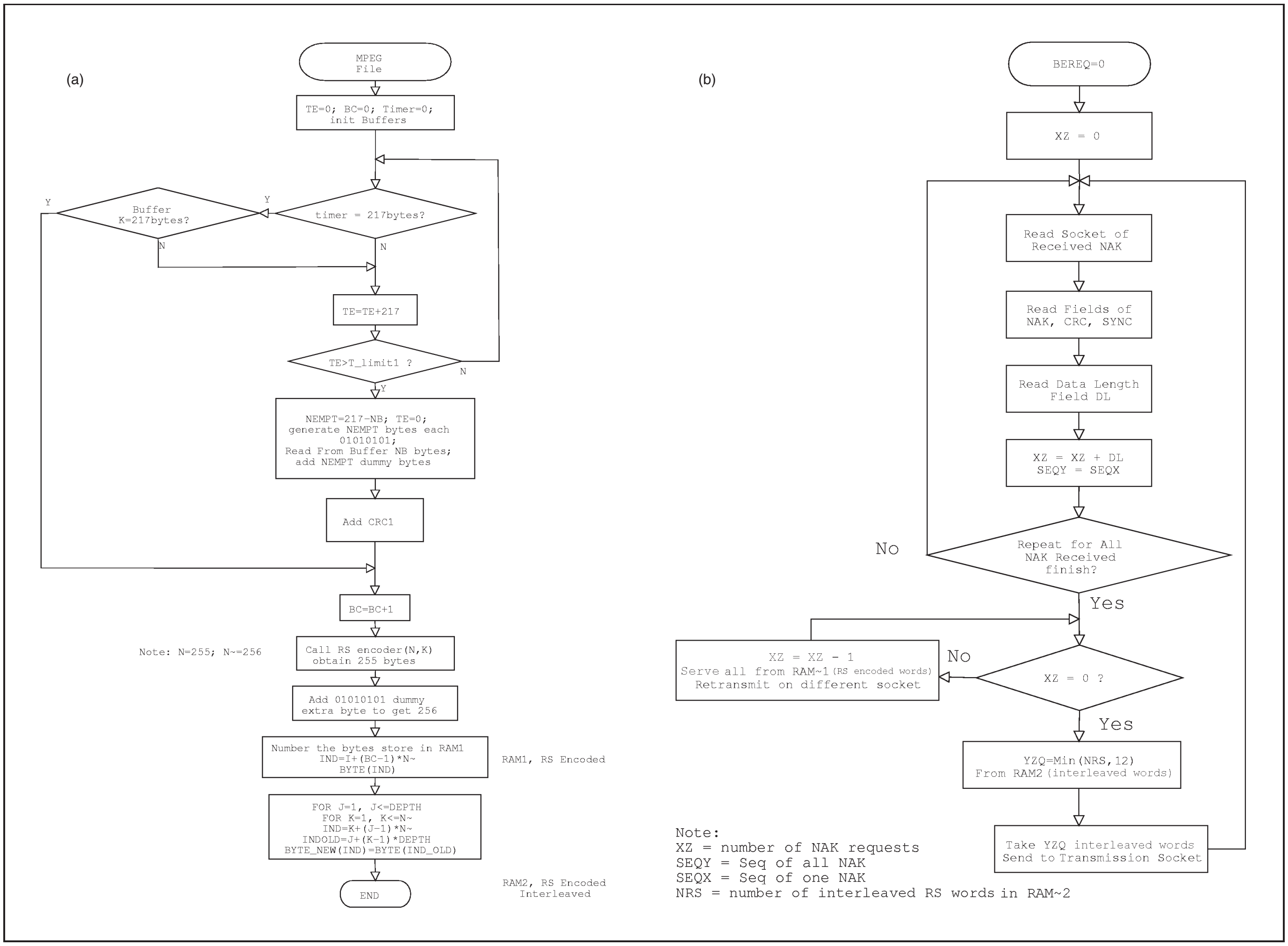

Figure 2: *(a) Source real-time encoding at server. (b) Transmission duty cycle at server.*

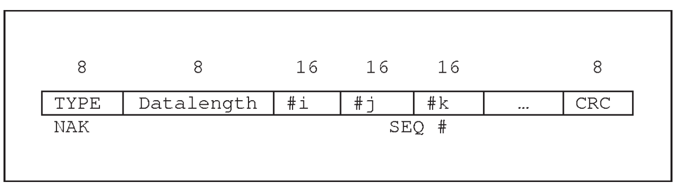

Figure 3: *NAK packet format.*

of bytes (at least $(K + E)$) of the same RS word are received. Only two NAKs or repair requests are allowed for the same RS word. The processing of the received repairs will be similar to the above, with the exception that no RS decoding takes place. Finally, as to video or voice applications, if all error correction trials for a certain RS word fail and playing time for this word expires, the previous correctly received 217 bytes in the receiver's buffer should be replayed. It is also possible that too many data arrive at the client. In this case, due to being busy with de-interleaving and RS decoding, the client may suffer from data loss.

To protect against the data loss introduced by processing constraints, flow control may be adopted. Because this approach is not very feasible in view of the underlying UDP protocol, we leave it for future investigation. To free up more processing time for miscellaneous error concealment mechanisms in this work, all successive 217-byte blocks of words delivered to the receiving buffer at the client should be transferred into another playing buffer rather than being played immediately. This playing buffer has a bit length of three interleaving windows. Once the buffer is filled up, it is discharged to the receiver player.

Fig. 4(b) shows the step-by-step generation of the NAK request at the client. As shown in Fig. 2(b), we execute the processing for such a NAK message as it arrives at the server. Fig. 4(b) also shows the multiplexing of RS (interleaved) words and repair (retransmission) words in server transmission time. Priority is given to repair words, and interleaved words are transmitted when there are no repairs to be sent ($XZ = 0$).

The timing diagram of Fig. 5 shows a sequence of events taking place under the control of our new scheme for a typical scenario. Equal-length interleaving windows for data records $N, N+1, N+2$ are used. The end of the next window sets the time limit for playing received packets of the current record at the MPEG player. Since received packets of each record are reorganized in the interleaving window and then played in sequence, this technique has a buffer effect that can help to guarantee delay-jitter-free playing at the receiver. This figure also shows the NAK and retransmissions for record N due to the packet loss. For record $N+1$, the RS FEC decoder can correct the loss and errors without asking for any retransmission. For record $N+2$, no loss and errors are encountered for the first K bytes of RS symbols; therefore the RS decoder is not called, and no NAKs are generated.

III. Experimental results

Fig. 6 shows a simple outline of our experimental environment. Although seven PCs are involved in the experiment to build up multicast, only two PCs (600 Mhz Pentium II computers with 256 MB of RAM), one acting as server and the other acting as client, are mainly addressed

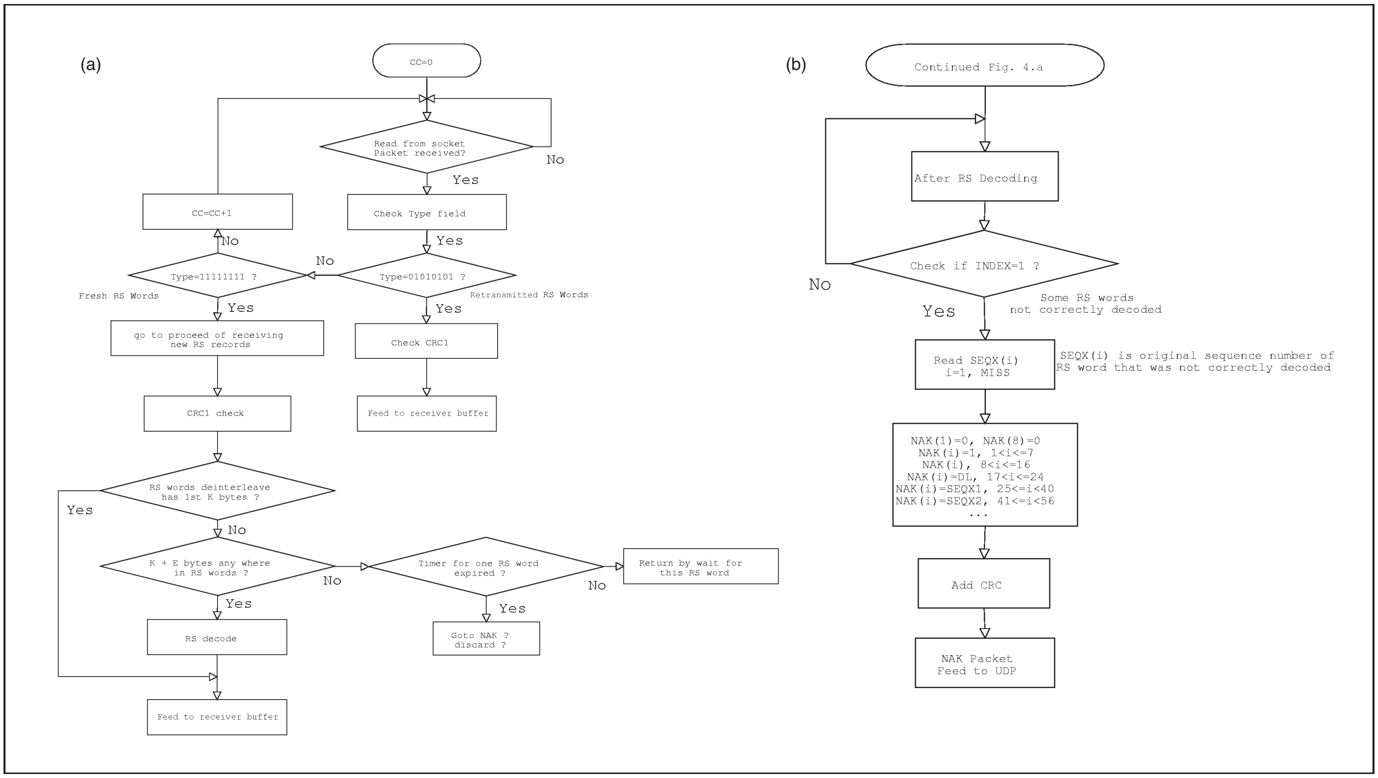

Figure 4: *(a) Receive at client. (b) Send-NAK packet at client.*

in Fig. 6. All PCs are connected by 100BaseT Ethernet virtual LANs, and a Linux real-time platform is run on them to evaluate the performance of proposed error control techniques. The IP Wave Network Impairment Emulator (IP Wave) software package on a Windows NT platform is configured as the default router between different virtual LANs to emulate the error and loss effects between server and clients. The IP Wave can simulate effects that may be encountered by any application communicating across a large network such as the Internet. The IP Wave includes a comprehensive set of applicable, real-world impairments such as packet delay, duplication, reordering, fragmentation, loss, and errors that can affect IP packets coming from distant locations or across distributed networks. In our experiment, the IP Wave is employed mainly to achieve various packet loss and error conditions (in Figs. 7–11) without requiring us to set up distributed endpoints geographically for testing.

Many tests have been conducted for different packet loss and byte errors. The main results obtained in our experiment are shown in Figs. 7 to 11. Fig. 7 shows the percentage of words for which the RS decoder is called according to the error detection by CRC. For low data rates, and/or low random loss, and/or low byte errors, this percentage is low but steadily rises as the loss and error increase. Moreover, under heavy packet loss and byte-error conditions, the RS decoder is called at an almost constant percentage to counteract the transmission inadequacy. From Fig. 8, we can see that the percentage of generated NAK messages does not increase as quickly as the calling rate of the RS decoder in Fig. 7, due to the fact that many damaged words are corrected by the erasure-correction capability of RS codes. Fig. 9 shows that with the use of our error control scheme, few retransmissions occur even at high data-rate, error, and loss probabilities. However, the ARQ mechanism is still indispensable, because sometimes the loss and errors are outside the scope that FEC can handle. Fig. 10 demonstrates the percentage of those words delivered correctly to the receiver buffer before the final time out. The overall efficiency and improvement obtained by the proposed interleaved FEC/ARQ algorithms can be seen clearly. By subtracting the values in Fig. 10 from 100%, we get the residual errors in Fig. 11. The residual errors reflect the percentage of those words that cannot be saved by any of the previously mentioned measures (ARQ, FEC, interleaving). In addition, the experimental results above show that the performance of our algorithms is sensitive mainly to packet loss and errors. Bandwidth expansion due to FEC/ARQ utilization may bring occasional congestion, packet loss and errors. However, test results indicate that the performance of our algorithms is not compromised as long as the CPU speeds of the server and clients are capable of handling the real-time interleaved FEC/ARQ coding/decoding algorithms therein.

IV. Theoretical analysis

In this section, we present the theoretical analysis of our interleaved FEC/ARQ scheme in a step-by-step way. First, we analyze the basic characteristics of our hybrid FEC/ARQ technique in unicast. Then we evaluate its performance in a multicast environment. We present a comparative performance analysis of the hybrid FEC/ARQ scheme and the ARQ-only scheme. We evaluate the performance enhancement of our scheme in terms of average number of transmissions and average number of NAKs generated. We also show the effect of varying sending rates, group sizes, and packet loss rate. Only packet loss, but not random error, is considered in this section.

A. Basic characteristics of hybrid FEC/ARQ in unicast

In this section, we assess the performance of the interleaved FEC/ARQ scheme in unicast, using the following notations:

P : Probability of packet loss.

I : Number of packets lost.

NN : Number of interleaved RS words per packet; $\mathrm{NN} = 8$.

N : Total number of packets in an interleaved-RS-words block (32).

K : Number of packets containing original data in an interleaved-RS-words block (28).

P' : Probability that a packet is incorrectly RS decoded.

$P_{\mathrm{no\,NAK}}$: Probability that no NAK is generated for a certain block at the receiver.

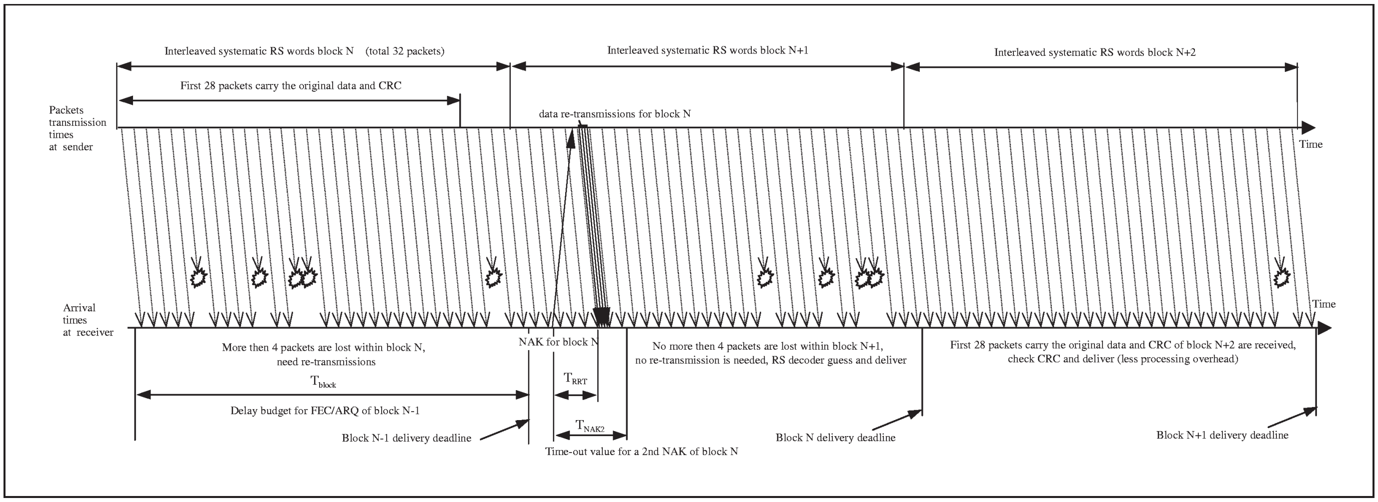

Figure 5: *The hybrid FEC/ARQ system timing.*

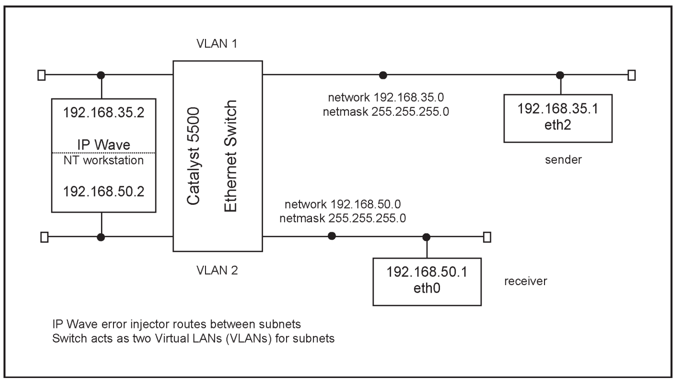

Figure 6: *Block diagram of the experimental test bed used (IP Wave used for error and loss injection).*

P_{NAK} : Probability that a block generates a NAK at the receiver (FEC/ARQ).

$P_{\mathrm{error\ RS/ARQ}}$: Expected residual packet error for the FEC/ARQ scheme.

$P_{\mathrm{error\ ARQ}}$: Expected residual packet error for the ARQ-only scheme.

Each packet contains eight interleaved RS words, so that the loss of one packet implies the loss of only eight symbols in each of 256 RS-encoded words. Since $(255, 223)$ Reed-Solomon codes can correct 32 erasures per RS word, the loss of four packets will lead to 32 erasures in each de-interleaved RS word, which is a correctable level of loss. When the number of packets lost is greater than four, a NAK will be generated. The sender will retransmit interleaved words containing only original data (no parity bytes). Equation (1) gives the probability that a packet cannot be RS decoded correctly and has to be retransmitted. Note that, as mentioned above, if 0 or 1 or 2, 3, 4 (less than $(N-K)$) packets are lost, then 0 or 8 or 16, 24, 32 bytes will be lost in the de-interleaved RS words, and there are no final decoding errors:

$$P' = \frac{1}{N} \sum_{i=N-K+1}^{N} i \binom{N}{i} (1-P)^{N-i} P^i. \tag{1}$$

For a systematic-RS-words block, no NAK is generated when the number of packets lost is less than $(N-K)$. Therefore, the probability that no NAK is generated for a block is

$$P_{\mathrm{no\ NAK}} = \sum_{i=0}^{N-K} \binom{N}{i} (1-P)^{N-i} P^i. \tag{2.a}$$

The probability that a block generates a NAK to one or more packets at one receiver is

$$P_{\mathrm{NAK}} = 1 - P_{\mathrm{no\ NAK}}. \tag{2.b}$$

If we limit the number of NAKs to two, the expected residual error for one packet in the FEC/ARQ scheme is

$$P_{\mathrm{error\ RS/ARQ}} = P' \cdot P^2. \tag{3}$$

This means that one RS decoding trial and two retransmission trials never result in final packet reception. Without FEC, the residual error rate is

$$P_{\mathrm{error\ ARQ}} = P^3. \tag{4}$$

This means that one transmission and two retransmissions never result in final packet reception either. Constrained by the delivery deadline, a NAK will be generated no more than twice for every interleaved block. For the same reason, a retransmission contains original data only, which means that after receiving retransmitted packets, no RS decoder is called. These assumptions apply throughout this paper.

B. Performance evaluation for group communication

Group communication performance is evaluated for the hybrid FEC/ARQ scheme in this section. In order to simplify the performance assessment of hybrid FEC/ARQ, we make several assumptions about the network model. We assume that link loss rates are not affected by the rate of the sender. This assumption is reasonable in a scenario where the congested links utilized by the protocol are also utilized by many other sessions. We assume that only the sender transmits repairs, and that these repairs are always sent to receivers by multicast. We consider the case where all the receivers have the same loss rate. NAKs are assumed to be sent using unicast only, and the NAK feedback from the various users is aggregated. When some subtrees share common losses, and no local loss exists in these subtrees, they are treated as one member when the equivalent group size with independent loss is counted. These assumptions make the system model simpler to evaluate. The additional notations used in this section are described as follows:

U : Independent-loss equivalent group size.

P'_{NAK} : Probability that a block generates a NAK a second time at the receiver (FEC/ARQ).

α : Probability that sender needs to retransmit a packet to any group member (FEC/ARQ).

β : Probability that sender needs to retransmit a packet to any group member (ARQ-only).

$E(\mathrm{TRAN}_{\mathrm{RS/ARQ\_group}})$: Average total transmission time per packet in the group (FEC/ARQ).

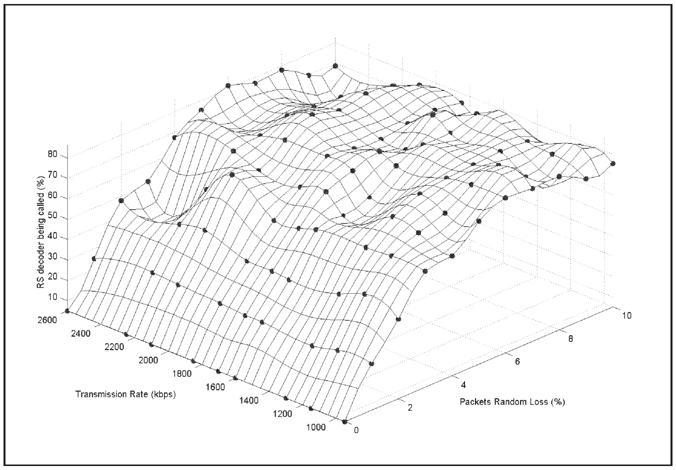

Figure 7: *Percentage of words for which RS decoder is called vs. rate and loss for the RS-ARQ hybrid scheme (6% RS words with byte errors).*

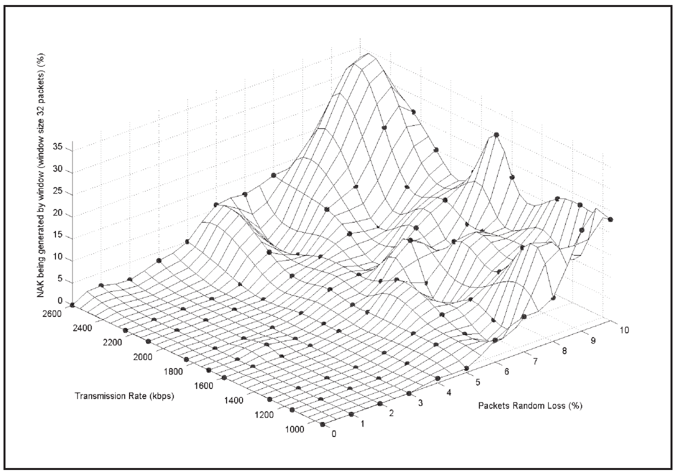

Figure 8: *Percentage of NAKs being generated vs. rate and loss for the RS-ARQ hybrid scheme (6% RS words with byte errors).*

$E(\mathrm{TRAN}_{\mathrm{ARQ\_group}})$: Average total transmission time per packet in the group (ARQ-only).

$E(\mathrm{NAK}_{\mathrm{RS/ARQ}})$: Average total NAKs generated per block in the group (FEC/ARQ).

$E(\mathrm{NAK}_{\mathrm{ARQ}})$: Average total NAKs generated per block in the group (ARQ-only).

While $(1-P')^U$ means that none of the group members require retransmission of a packet, the probability that the sender will need to retransmit a packet to any group member for the first trial in the FEC/ARQ scheme is

$$\alpha = 1-(1-P')^U. \tag{5}$$

The average number of receivers needing retransmission is $P' \cdot U$. The probability of packet loss in retransmission is P. Therefore, we can make a useful approximation for the probability that a sender will need to retransmit a packet a second time to any group member that may need it for the FEC/ARQ scheme:

$$\alpha' \approx 1-(1-P)^{P' \cdot U}. \tag{6}$$

Similarly, the probability that a sender will need to retransmit a packet to any group member in the ARQ-only scheme is

$$\beta = 1-(1-P)^U. \tag{7}$$

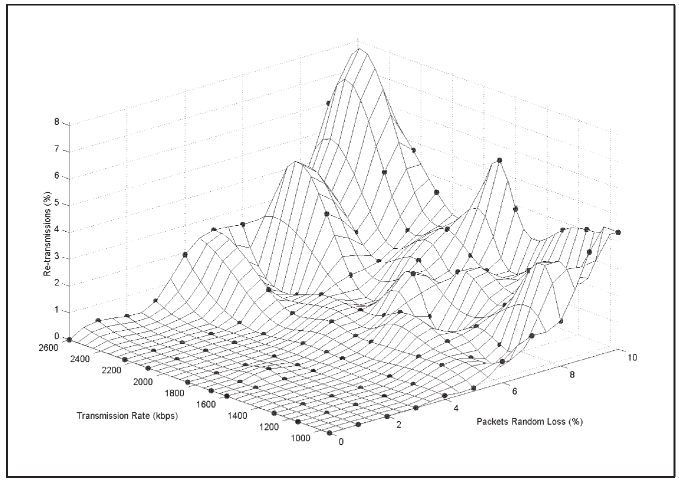

Figure 9: *Retransmission percentage vs. rate and loss for the RS-ARQ hybrid scheme (6% RS words with byte errors).*

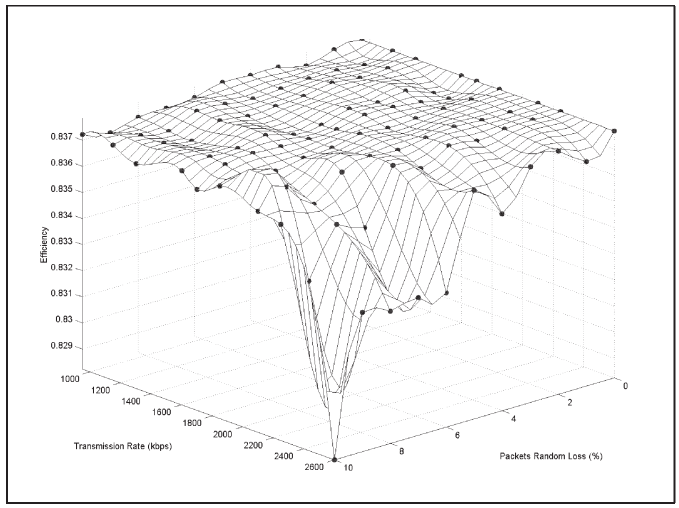

Figure 10: *Efficiency vs. rate and loss for the RS-ARQ hybrid scheme (6% RS words with byte errors).*

An approximation for the probability that a sender will need to retransmit a packet a second time to any group member that may need it for the ARQ-only scheme is

$$\beta' \approx 1-(1-P)^{P \cdot U}. \tag{8}$$

Considering the FEC/ARQ scheme, the first transmission of an interleaved RS block will take N packets. The retransmissions will have only the first K original data packets. In the second retransmission, the packet is accepted as is, whether it is right or wrong. Therefore, the average total transmission time per packet in the group for the FEC/ARQ scheme is

$$\begin{aligned} E(\mathrm{TRAN}_{\mathrm{RS/ARQ\_group}}) &= \frac{1}{K}\big((1-\alpha)N + \alpha(1-\alpha') \\ &\quad \cdot (N+K) + \alpha \cdot \alpha'(N+2K)\big). \\ \therefore E(\mathrm{TRAN}_{\mathrm{RS/ARQ\_group}}) &= \frac{N}{K} + \alpha + \alpha \cdot \alpha', \end{aligned} \tag{9}$$

where the first term corresponds to the success of the first trial, the second term indicates the first retransmission trial, and the third term represents the second retransmission trial. Similarly, the average total transmission time per packet in the group for the ARQ-only scheme is

$$\begin{aligned} E(\mathrm{TRAN}_{\mathrm{ARQ\_group}}) &= (1-\beta)\cdot 1 + \beta(1-\beta')\cdot 2 + \beta\cdot\beta'\cdot 3. \\ \therefore E(\mathrm{TRAN}_{\mathrm{ARQ\_group}}) &= 1 + \beta + \beta\cdot\beta'. \end{aligned} \tag{10}$$

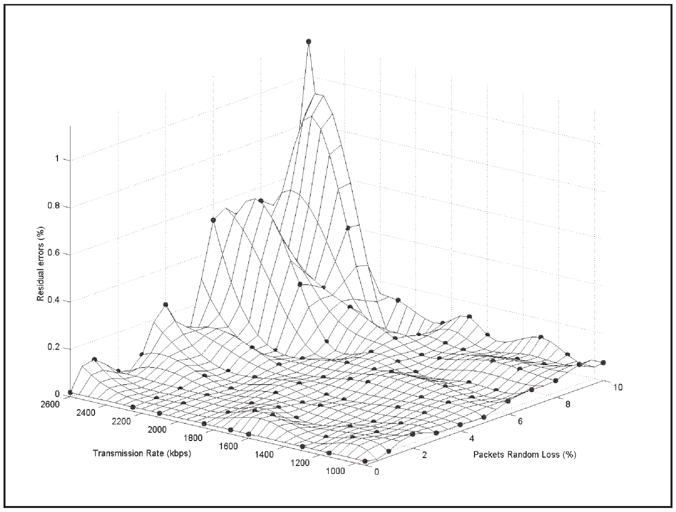

Figure 11: *Percentage of residual errors vs. rate and loss for the RS-ARQ hybrid scheme (6% RS words with byte errors).*

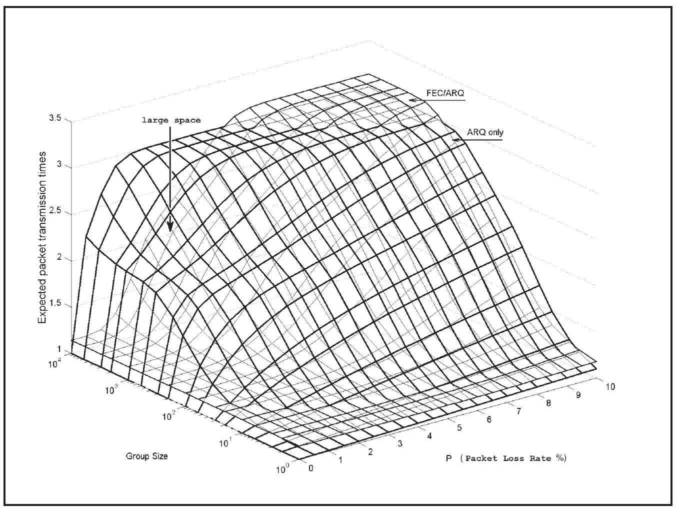

Figure 12: *Comparison of expected packet transmission times (hybrid FEC/ARQ vs. ARQ-only).*

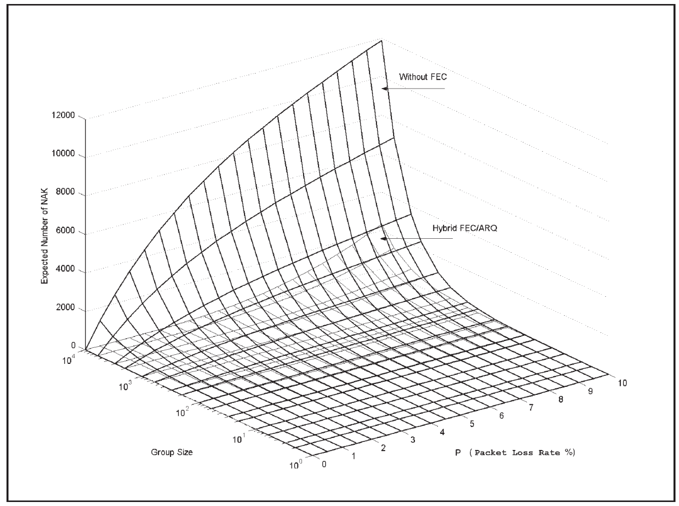

Figure 13: *Comparison of expected number of NAKs generated per window (hybrid FEC/ARQ and ARQ-only).*

Because no parity packets are transmitted, the N/K term in (9) corresponds to "1" in (10), while other terms in (10) carry interpretations similar to those in (9).

An approximation for the probability that a block will generate a NAK the second time at one receiver (FEC/ARQ scheme) is $P'_{\mathrm{NAK}} \approx 1-(1-P)^{K\cdot P'}$. Combining this P'_{NAK} and the P_{NAK} in (2), we derive the average total number of NAKs generated per block in the whole group for the FEC/ARQ scheme as $E(\mathrm{NAK}_{\mathrm{RS/ARQ}}) = (P_{\mathrm{NAK}} \cdot (1-P'_{\mathrm{NAK}}) \cdot 1 + P_{\mathrm{NAK}} \cdot P'_{\mathrm{NAK}} \cdot 2) \cdot U$, where the first term and the second term respectively account for the probability that one or two NAKs will be transmitted in the two trials.

$$\therefore E(\mathrm{NAK}_{\mathrm{RS/ARQ}}) = U \cdot P_{\mathrm{NAK}}(1 + P'_{\mathrm{NAK}}). \tag{11}$$

Similarly, using $P'_{\mathrm{NAK}} = 1 - (1-P)^{PK}$, we can approximate the average total number of NAKs generated per block in the group for the ARQ-only scheme:

$$E(\mathrm{NAK}_{\mathrm{ARQ}}) \approx U \cdot \left(1-(1-P)^{K}\right) \cdot \left(2-(1-P)^{PK}\right). \tag{12}$$

The following figures show some of the multicast FEC/ARQ performance results obtained. We assume a multicast group size ranging from 1 to 10 000 with independent loss; the packet loss rate ranges from 0 to 10%, which is a very common case for the Internet. Moreover, our hybrid FEC/ARQ scheme limits the number of NAKs to two. We compare the results of the hybrid FEC/ARQ scheme with those from other papers, lab results and the results of the ARQ-only scheme to determine the improvement achieved by using FEC. We choose the window size as 28 packets for the ARQ-only scheme; this size is equal to that of the interleaved RS block's original data part.

Fig. 12 shows the transmission times achieved by both the FEC/ARQ scheme and the ARQ-only scheme ((9) and (10)). First, the results of this figure are compared with the conclusion mentioned in [10]. In Fig. 1 of [10], the number of transmissions required to correctly transfer a single packet without any protective measures is given. The numerical results of the ARQ-only scheme shown in Fig. 12 match the results from Fig. 1 of [10] very well. To some extent, this validates the correctness of our assumptions and analytical results. Second, we compare the numerical results of the FEC/ARQ scheme in Fig. 12 with the lab results shown in Fig. 10 under the condition that the group size is 6. The values of the expected packet transmission times on the y-axis of Fig. 12 can be transformed to the values of efficiency on the y-axis of Fig. 10 by means of the formula $\mathrm{Efficiency} = 1/\mathrm{Expected\ Packet\ Transmission\ Time}$. After the transformation, we find that the values in Fig. 12 are a little lower than the lab results in Fig. 10 in terms of efficiency. This outcome is reasonable, because the values in Fig. 12 are for the worst case. Third, let us compare the values of the FEC/ARQ scheme and the ARQ-only scheme in Fig. 12. There is a large difference between transmissions of the two schemes as indicated, showing the coding power of FEC, especially when the loss rate is less than 4% and the group size is large. When packet loss rates are high and group size is large, the FEC/ARQ scheme needs more transmissions because of the parities sent the first time. But this does not necessarily mean that the ARQ-only scheme is better, because this small number of parities will contribute to the generation of fewer NAKs and a lower residual error rate.

Fig. 13 shows the strong effect of FEC in terms of minimizing NAKs even though the number of parities is small ((11) and (12), FEC rate $= 233/255$, not $1/2$). Moreover, this figure yields the average total number of NAKs generated per block in the hybrid FEC/ARQ scheme with group size from 0 to 10 000. If we transform the values of this figure by the formula $\mathrm{NAK\ Being\ Generated}\ (\%) = \mathrm{Expected\ Number\ of\ NAKs/Group\ Size}$, one may conclude that the results of the hybrid FEC/ARQ scheme in this figure match the lab results of Fig. 8 very well when the group size is 6 (small due to the lab space and costs).

In summary, comparing the two schemes, we find that FEC/ARQ performs better than ARQ-only in terms of requiring not only fewer transmissions, but also fewer NAKs. The hybrid FEC/ARQ scheme is applicable within a reasonable range of packet loss and group-size values.

V. Conclusion

In this paper, an interleaved FEC/ARQ scheme was proposed to provide good QoS experimental results when applied to multicast transmissions. While some theoretical results have been recently presented for miscellaneous reliable multicast techniques, our laboratory work establishes the practical feasibility of integrating all these techniques. According to the practical results of this test bed, one can send a video stream over the Internet under conditions of heavy loss and errors. Our experimental results indicate that the receivers can correct almost all the loss and errors while operating at a reasonable data rate, i.e., 1 Mb/s video transmission. Moreover, analysis results reveal that our interleaving FEC/ARQ scheme can work well when the group size is under 100 and packet loss rate is less than 10%. Currently, the experiments are being extended to teleconferencing applications.

Acknowledgements

This work was supported by grants from Bell Canada and from the Natural Sciences and Engineering Research Council of Canada.

References

[1] S.B. Vicker, *Error Control Systems for Digital Communications and Storage*, Upper Saddle River, N.J.: Prentice-Hall, 1995.

[2] 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.

[3] L. Rizzo and L. Vicisano, “Effective erasure codes for reliable computer communication protocols,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 2, Apr. 1997, pp. 24–36.

[4] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang, “A reliable multicast framework for light-weight sessions and application level framing,” *IEEE/ACM Trans. Networking*, vol. 5, no. 6, Dec. 1997, pp. 784–803.

[5] Christos Pappadopoulos, Guru Parulkar, and George Varghese, “An error control scheme for large-scale multicast applications,” in *Proc. INFOCOM’98*, vol. 3, Mar. 29–Apr. 2, 1998, pp. 1188–1196.

[6] J. Lin and S. Paul, “RMTP: A reliable multicast transport protocol,” in *Proc. IEEE Infocom’96*, Mar. 1996, pp. 1414–1424.

[7] J.H. Jeng and T.K. Truong, “On decoding of both errors and erasures of a Reed-Solomon code using an inverse-free Berlekamp-Massey algorithm,” *IEEE Trans. Commun.*, vol. 47, no. 10, Oct. 1999, pp. 1488–1494.

[8] J.X. Yu, Li Yuan, H. Murata, and S. Yoshida, “Hybrid-ARQ scheme using different TCM for retransmission,” *IEEE Trans. Commun.*, vol. 48, no. 10, Oct. 2000, pp. 1609–1613.

[9] M.A. Kousa, A.K. Elhakeem, and H. Yang, “Performance of ATM networks under hybrid ARQ/FEC error control scheme,” *IEEE/ACM Trans. Networking*, vol. 7, no. 6, Dec. 1999, pp. 917–925.

[10] D.A.M. Villela and O.C.M.B. Duarte, “Improving scalability on reliable multicast communications,” *Comput. Commun.*, vol. 24, no. 5/6, Mar. 2001, pp. 548–562.