Multimed Tools Appl (2006) 29: 305–323
DOI 10.1007/s11042-006-0016-4

Generic forward error correction of short frames for IP streaming applications

Jari Korhonen · Yicheng Huang · Ye Wang

Published online: 16 June 2006

Abstract If the frame size of a multimedia encoder is small, Internet Protocol (IP) streaming applications need to pack many encoded media frames in each Real-time Transport Protocol (RTP) packet to avoid unnecessary header overhead. The generic forward error correction (FEC) mechanisms proposed in the literature for RTP transmission do not perform optimally in terms of stability when the RTP payload consists of several individual data elements of equal priority. In this paper, we present a novel approach for generating FEC packets optimized for applications packing multiple individually decodable media frames in each RTP payload. In the proposed method, a set of frames and its corresponding FEC data are spread among multiple packets so that the experienced frame loss rate does not vary greatly under different packet loss patterns. We verify the performance improvement gained against traditional generic FEC by analyzing and comparing the variance of the residual frame loss rate in the proposed packetization scheme and in the baseline generic FEC.

Keywords Forward error correction (FEC) · Interleaving · Multimedia streaming · Real-time transport protocol (RTP)

1 Introduction

Multimedia streaming is rapidly gaining popularity in the world of IP networking. In large bandwidth wireline packet networks, Internet telephony (VoIP) and Video-on-Demand (VoD) are already a part of consumers' everyday life. In the wireless domain, the change from traditional data communications toward real-time content delivery is in progress.

J. Korhonen · Y. Huang · Y. Wang (✉)
Department of Computer Science, School of Computing, National University of Singapore,
3 Science Drive 2, Singapore 117543, Singapore
e-mail: wangye@comp.nus.edu.sg

J. Korhonen
e-mail: jari.ta.korhonen@nokia.com

Y. Huang
e-mail: huangyic@comp.nus.edu.sg

The requirements for real-time data transport mechanisms are distinctively different from those for traditional data communications: the transport rate is typically fixed to the bit rate of the multimedia encoder and the system cannot change the transmission rate according to varying network throughput as flexibly as in applications using Transmission Control Protocol (TCP). More importantly, real-time delivery requirements restrict the use of retransmissions to recover from packet losses. This is why streaming applications must often rely on reproducing missing data with purely receiver-based error concealment or redundancy-based transmission strategies, which are often known as forward error correction (FEC).

In receiver-based error concealment schemes, the data transport system and protocols are not involved in packet loss recovery. Error concealment is based purely on the subjective characteristics of the multimedia data. In this kind of error concealment, missing data sections are reproduced using, for example, data repetition or interpolation. Readers interested in receiver-based error concealment may refer to the survey on error recovery schemes for audiovisual data by Wah et al. [17] or for audio by Perkins et al. [12].

The most efficient receiver-based error concealment techniques are codec-dependent or mathematically complex, requiring high processing power. More desirable trade-off between complexity and subjective quality can be achieved by combining lightweight error concealment with simple packet loss recovery schemes based on either retransmissions or FEC.

RTP, specified in RFC 3550 [15], is the de facto standard for delivering data with real-time content over IP networks. The initial RTP design favors multicast-oriented applications with strict real-time constraints, such as multiparty conferencing and live broadcasting. However, the range of streaming applications is wide, allowing video- and audio-on-demand applications with more relaxed real-time requirements to benefit from retransmissions. This is why a payload format supporting selective retransmissions using RTP has been proposed [13]. In any case, retransmission-based packet loss recovery in real-time communications is not always an option. For instance, retransmissions in multicast applications could cause problems that are difficult to solve, such as feedback implosion [7]. Although protocols for reliable multicast have been proposed, they do not provide an appealing solution for real-time applications.

Forward error correction can be used even if there is no feedback channel at all. In fact, FEC is typically the best choice for real-time multicast applications or real-time networking in general when round-trip times are long. FEC schemes can roughly be classified into two categories: application specific and generic. Application specific FEC schemes are designed specifically for certain multimedia codecs, with particular attention on the priority of each bit. Generic FEC is codec independent.

The existing generic FEC methods in the literature are generally designed for recovering full RTP packet payloads or the priority sections in each payload. The main problem of such an approach is unstable performance. If the FEC scheme fails to recover an RTP payload, all data included in the RTP packet is lost, even if there are multiple individual frames. At the same time, the observed data loss rate varies greatly according to the packet loss pattern. This is because the actual data loss rate does not depend only on the packet loss rate, but also on *which* packets are lost. To the best of our knowledge, we are the first to address the issue that the effective unrecoverable transient frame erasure rate can vary even with constant packet loss

rate. This issue is relevant because from the viewpoint of the end user, slight but stable degradation of audio or video quality is more acceptable than sudden fluctuations in quality.

In this paper, we propose an alternative strategy for generating RTP payloads consisting of multiple individual short data units (frames) and their corresponding FEC units. In our scheme, FEC data is computed for frames, not RTP payloads. Since each RTP packet carries several frames, we can efficiently decrease the variation in frame loss rates under different packet loss scenarios by shuffling media frames and FEC frames among RTP payloads efficiently. The performance improvement can be shown by comparing the proposed scheme against the baseline generic FEC in terms of the variance in residual frame loss rate under different packet loss scenarios.

2 Background and related work

2.1 Packetization and scheduling

Frame sizes are highly dependent on the audio or video encoder being used. In video coding, there are typically many different types of frames, and the frame size varies much as well. Because there is an upper limit for transport unit (packet) size in IP networks, large frames must be fragmented. In audio and speech coding, fragmentation is not usually needed because the frame size is typically much smaller than the maximum packet size.

The normal frame size of a high quality audio codec is around 400 B; for example, MP3 (128 kbit/s stereo, 44,100 Hz sample rate) uses the constant frame size of 418 B, and the corresponding MPEG Advanced Audio Codec (AAC) bitstream comprises variably sized frames with an average length of around 380 B [6]. Speech codecs use very small frames; for example, Wideband Adaptive Multi-Rate (AMR-WB) speech frame size at 23.85 kbit/s is fixed at 61 B, and the frame size for AMR at 4.75 kbit/s is only 14 B [5].

Each protocol layer appends some header data to the packets, causing significant header overhead if the payload is small. Robust header compression (ROHC) [1] can be used to reduce the size of the IP/UDP/RTP headers. However, not even ROHC can fully solve the problem because link and physical layer headers as well as a compressed version of the IP/UDP/RTP header still remain. This is why it is often necessary to pack several frames in one RTP packet even if ROHC is used. There are two major drawbacks in packing several frames in one RTP packet. First, it makes the RTP stream more vulnerable because every packet loss erases several clustered frames. Second, it causes extra delay at the sender, as the packetizer needs to wait for all the frames from the encoder to arrive before the RTP packet can be assembled and transmitted.

There are several approaches to tackling the first problem. The traditional method is to use interleaving to spread adjacent frames in different packets [11]. However, interleaving introduces an extra delay, which makes the second problem even more serious. The severity of the latency problem depends much on the application. Generally, packetization latency and interleaving delay are concerns only in highly interactive applications, such as Internet telephony. Even in this kind of applications, observed end-to-end delays of up to 400 ms are usually acceptable.

Because the typical length of a speech frame is around 20 to 30 ms, at least some frames can be packed together.

2.2 Generic FEC

Generic FEC is a codec independent method of protecting RTP payloads against packet erasures by adding redundant data to the transport stream. There are several variations of generic FEC. In its most trivial form, original data is replicated and transmitted several times to increase the probability that at least one of the copies is received. The drawback of this simple replication is the enormous proportional overhead. However, even the use of a small amount of redundancy is sufficient to achieve an adequate level of error protection.

A common method for generating FEC data is to take a set of packet payloads and apply the binary exclusive or (XOR) operation across the payloads. This scheme allows the recovery of missing data in the case where one of the original packets is lost but the FEC packet is received correctly. The RTP payload format for using generic FEC based on XOR operations has been published in RFC 2733 [14].

There are also many other more complex error correcting codes. Typically, these codes are designed for detecting and correcting bit errors instead of data erasures. However, in recent years, several proposals have been made to use well-known error correcting codes, such as Reed-Solomon codes, for packet loss recovery as well [14].

In the perspective of a transport protocol, all these codes work quite similarly: a set of original n data packets are protected by k parity packets, resulting in $n + k$ packets in total. If any n of these $n + k$ packets (including parity packets) are received, all the lost original packets can be recovered [10]. The weakness of the more complex schemes is computational complexity, which may cause performance problems with long packets and large values of k and n. This is why we limit the scope of this paper to XOR-based FEC codes only. However, the basic principles discussed are easily convertible for other kinds of linear codes.

Figure 1 shows two basic schemes using the generic FEC defined in RFC 2733. In this paper, we adopt the definition of function f(x, y,...) to denote the resulting FEC packet when the XOR operation is applied to the packets x, y... from RFC 2733. In example (a), every single packet loss in the original media stream can be recovered, and in example (b), every packet loss can be recovered, assuming that the FEC stream is received correctly in both cases. However, both schemes require more

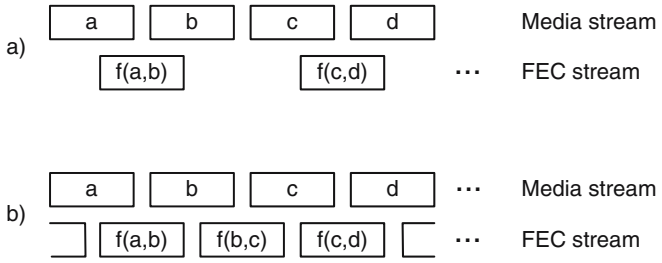


Fig. 1 Two sample schemes using generic FEC

network bandwidth because of the redundancy overhead: 50% in case (a) and 100% in case (b). More examples are given in RFC 2733.

In practice, the media stream and the FEC stream are usually transmitted using the same transport medium. This is why we cannot expect packet losses to occur only in the media stream as both streams are likely to suffer from similar error characteristics. In the network perspective, it is realistic to assume the media stream and the FEC stream to form a single stream containing both media and FEC packets. Given a sequence of media and FEC packets, we can easily see the variation in error recovery rates when we examine the residual media data loss rate after applying different kinds of loss patterns to the sequence.

2.3 Interleaving

According to several studies, bursty losses of multimedia frames or blocks are much more harmful to perceived quality and more difficult to conceal than smaller gaps. The same applies to speech, audio, video and even still images [2, 3, 9, 21]. Interleaving is commonly used in multimedia streaming to avoid losses of clustered frames or data blocks. Basically, there are two forms of interleaving commonly used. If several blocks are packed in one packet, the interleaver allocates adjacent blocks to different packets (block interleaving). If each packet only contains one block, the interleaver rearranges packets so that adjacent packets are not transmitted consecutively (packet interleaving). In contrast to block interleaving, packet interleaving is beneficial only in the presence of bursty packet losses.

There are several examples of techniques involving block interleaving. One such scheme for audio streaming has been presented in [6], where the spectral elements of audio frames are spread among several RTP packets to facilitate error concealment. In that scheme, the critical header data may be extracted from the audio frames and packed separately in different packets to be transported using more reliable means [18]. Similar strategies have also been developed for video streaming, where video frames are split up into smaller elements (blocks or slices) [16, 19, 20]. This kind of strategies can be combined with Unequal Error Protection (UEP) schemes by applying stronger FEC to high priority data [19].

Optimal mode and depth of the interleaving cycle depend on the application and network conditions. Typical packet loss bursts in the Internet are rather short. According to the measurements by Loguinov and Radha, the average burst length is about two packets [8]. Most of the proposed interleaving schemes for IP networking rely on predefined parameters. There are some proposals for adaptive interleaving, but mostly concerning radio access networks rather than IP networks [2].

3 Generic FEC for short blocks

The generic FEC reviewed in the previous section does not cater for RTP payloads consisting of several individually decodable multimedia frames or data blocks. In this section, we propose methods to extend the use of generic FEC to RTP payloads containing separate frames or blocks. For brevity, we refer to all these elements as ‘frames.’

3.1 Problem formulation

In theory, the remaining packet loss probability after using simple XOR-based FEC follows Eq. 1, where p_{res} is the residual packet loss rate, p is the packet loss rate in the communications channel, and n is the number of packets in the set covered by the FEC computation.

$$p_{res} = p(1 - (1 - p)^n) \tag{1}$$

However, the equation is valid only if the packet loss rate is measured over a long sequence of packets. The local frame loss characteristics depend highly on the exact packetization method. Considering a sequence of packets protected by one FEC packet, the observed packet loss rate does not depend on the number of lost packets only, but also on the packet loss pattern. One packet lost within a sequence can always be recovered. If two packets in the same sequence are lost, we need to make a distinction between two cases: two data packets are lost, or one data packet and the related FEC packet are lost. In the first case, two original media packets are lost. In the latter case, residual data loss comprises only one original media packet.

This is an undesirable feature because the same number of RTP packets lost may lead to different numbers of residual packet losses observed above the RTP protocol layer. This increases fluctuation in local residual media frame erasure characteristics even under a constant packet loss rate. In multimedia streaming, the quality perceived by the end user drops especially in the case of occasional heavily clustered data losses. More stable performance would guarantee better subjective output, even when the overall data loss rate remains the same.

When there are multiple frames in each packet, the method of arranging frames into packets needs to be considered carefully because each packet that is not recovered leads to several frames being erased. To allow as stable a performance as possible, the local residual frame loss rate should depend on the packet loss rate only, not the pattern of which specific packets are lost. Thus, the problem is to find an optimal allocation of frames in packet payloads that fulfills this condition. The essential measure for the quality of packet allocation is the variance in the frame loss rate when only the packet loss pattern changes while the packet loss rate over the sequence of a certain number of packets remains the same.

3.2 Definitions and outline for stable packetization

Assume that when there are frames x, y,... and a related parity FEC frame f(x, y,...), these data frames and the FEC frame form a group of mutually dependent frames. This definition is justified by the fact that a lost frame can be recovered by applying the XOR operation to all the remaining dependent frames. An example is shown in figure 2, where all mutually dependent frames are connected with lines. In general, when the XOR operation is applied to n frames, the total number of frames in the group (including the FEC frame) is $n + 1$. When n is small, error protection is strong but redundancy overhead is large, as theoretical residual frame loss follows Eq. 1 and redundancy overhead follows $1/n$.

In the following, it is assumed that there is a fixed number of slots, s, in every RTP packet payload to be occupied by frames. In practice, s must be chosen so that it is smaller or equal to the maximum packet payload size divided by the frame size.

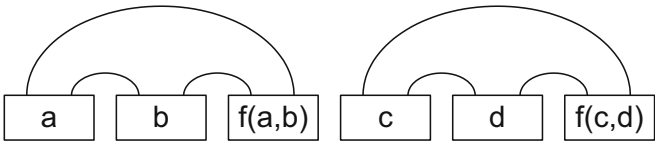


Fig. 2 Dependencies between data and FEC packets

For simplicity, we assume that all frames are equal in size. If this is not the case, the length of each frame has to be indicated within the additional data inserted in each packet.

As long as the maximum payload size is not exceeded, s can be chosen relatively freely. However, a small s and a small frame size would cause significant header overhead; this is why it is usually reasonable to choose s close to the maximum. If possible, s should also be chosen so that it is a multiple of $n + 1$. This would allow FEC frames and media frames to be spread evenly among all packets.

In an ideal situation, the number of frames and packets in an interleaving cycle cannot be selected arbitrarily, but they must fulfill certain conditions. First of all, P packets should accommodate exactly N FEC groups of packets (a FEC group is a group of mutually dependent frames, such as {a, b, f(a,b)}). No slots should be left empty. Therefore, condition (2) should be valid.

$$N(n+1) = Ps \tag{2}$$

In addition, there should be exactly as many dependencies between packets as there are dependencies between the frames to be allocated in these packets. Every FEC group contains n media frames plus one FEC frame. Thus, there are $(n^2 + n)/2$ pairs of mutually depending frames in each FEC group, resulting in $N(n^2 + n)/2$ pairs of mutually dependent frames in total, where N is the number of FEC groups. Given P packets, there are $(P^2 - P)/2$ unique pairs of packets in total. In an optimal situation, $N(n^2 + n)/2$ pairs of frames can be allocated evenly among $(P^2 - P)/2$ pairs of packets. In this case condition (3) applies. All the variables should have integer values to fulfill the conditions.

$$N\left(n^2 + n\right) = P^2 - P \tag{3}$$

From Eqs. 2 and 3, it is possible to solve the optimal number of frames F, packets P and groups of frames N for each cycle with Eqs. 4–6, respectively. If these conditions are not met, frames cannot be allocated in packets optimally, i.e., some slots are left empty or parallel dependencies exist between packets.

$$P = sn + 1 \tag{4}$$

$$F = Ps \tag{5}$$

$$N = F/(n+1) \tag{6}$$

For example, if there is room for four frames per packet ($s = 4$) and the XOR operation is applied to a set of three frames ($n = 3$), the optimal number of packets (P) is 13, containing 52 frames (F), of which 39 are original media frames and 13 FEC frames. Table 1 summarizes some useful parameter combinations. Note that a

Table 1 Some useful parameter combinations

n	*s*	*P*	*F*	*N*	FEC overhead (in %)
2	3	7	21	7	50
2	4	9	36	12	50
2	6	13	78	26	50
3	4	13	52	13	33
3	8	25	200	50	33
4	5	21	105	21	25

large value of s is better in terms of header overhead reduction, but a larger s also leads to a larger F and a larger N, increasing the delay in packet generation and reassembly processes.

In this paper, we consider IP networks where only packet erasures are concerned. Therefore, all the dependencies between frames located in different packets are actually dependencies between packets. Obviously, the loss of a packet pair with several redundant mutual dependencies results in more frame losses than the loss of a packet pair containing only a single pair of mutually dependent frames. In other words, redundant dependencies should be effectively avoided because heavily biased dependencies lead to unstable performance. This is shown in figure 3, where dependencies between packets using the traditional generic FEC are marked with lines.

In this example, one packet loss can always be recovered entirely. However, the loss of two out of the first three packets could lead to loss of three or six media frames. If the FEC frames are spread evenly among payloads, the situation is changed: as shown in figure 4, loss of two packets out of the first three packets would then always lead to loss of four frames. This is a clear improvement to the baseline generic FEC scheme. However, the interleaving effect is still limited due to the short interleaving cycle with three packets only. Much more significant performance improvement can be achieved if a longer interleaving cycle is involved, but optimal frame allocation would no longer be a trivial problem.

Following our intuition, we can easily draw up the basic guidelines for packetization. First of all, frames that are mutually dependent must not be located in the same packet. Second, redundant dependencies between the same packets should be avoided. The problem of optimal packetization can then be formulated as follows: every group of $n + 1$ mutually dependent frames should be allocated in the

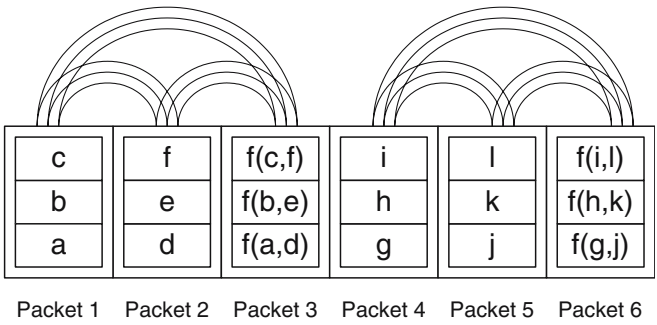


Fig. 3 Baseline generic FEC and the dependencies between packets

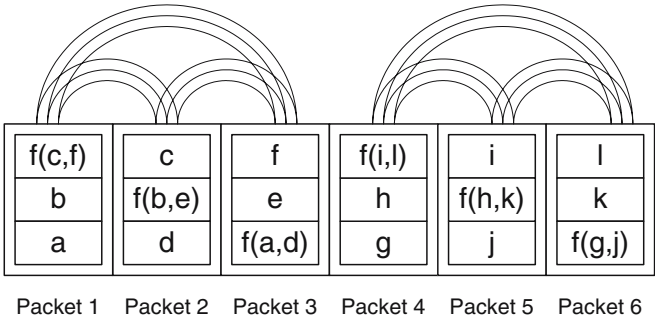


Fig. 4 Generic FEC with even spread of FEC frames among packets

packets so that there are no two (or more) groups with any two (or more) frames located in the same packet. In other words, each group of mutually dependent frames has to be allocated in a unique set of packets that is not overlapping any other set of packets by more than one packet. Another important principle is to spread FEC frames evenly across payloads. This can be achieved if the number of slots s in each payload is a multiple of the number of frames per FEC group.

A packetization example following the principles presented above is illustrated in figure 5. In this example, $s = 3$ and $n = 2$, when the resulting number of packets and frames are $P = 7$ and $F = 21$ (of which 14 are media frames and seven are FEC frames). Each of the groups of dependent frames has its own unique combination of packets not conflicting with any other packet combination. In this example, the packet combinations are (1,2,3), (1,4,5), (1,6,7), (2,4,6), (2,5,7), (3,4,7) and (3,5,6). Therefore, the first group containing frames a, b, and f(a,b) is allocated to packets 1, 2 and 3; the second group of frames c, d, and f(c,d) to packets 1, 4 and 5, etc. As we can see, there are no redundant dependencies between the same packet pairs as in the example in figure 3. Another advantage of the proposed scheme that can be seen in figure 5 is the interleaving effect. Because adjacent media frames are allocated in different packets, bursty frame losses are easier to avoid. Also, this facilitates the concealment of missing frames.

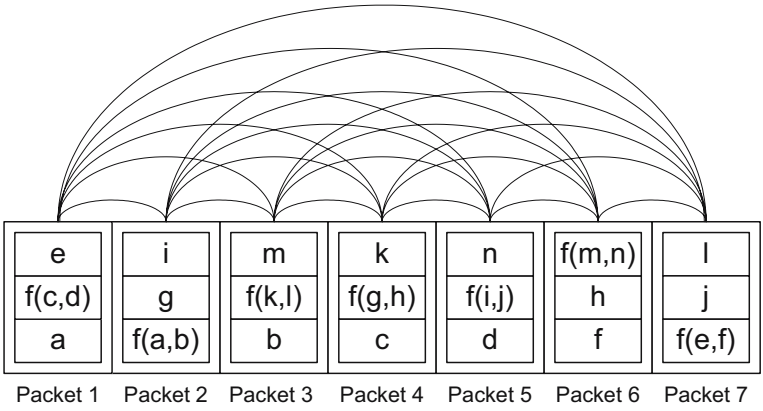


Fig. 5 A packetization example following the proposed scheme

Finding an optimal allocation when $n = 2$ and $s = 3$ is a special case of the three-dimensional matching problem, which has been proven to be NP-complete in a strong sense [4]. This is why the frame allocation problem can be solved only in polynomial time. Expectably, the problem is at least as difficult with larger values for n and s. However, for relatively small values of s and n, it is reasonable to use a computer program to systematically try all the possible frame allocation combinations to find the possible solutions. The result can be used to generate static frame allocation tables for the streaming application. For this purpose, we have implemented the recursive algorithm described below. In the streaming application perspective, the frame allocation is known *a priori* and the complexity of the algorithm is not an issue.

We say that two packet combinations conflict if they contain two or more packets that are the same. For example, combinations (1,2,3) and (1,2,4) conflict, but (1,2,3) and (1,4,5) do not. In the algorithm presented, the system computes all the different packet combinations in a logical order: (1,2,3), (1,2,4), (1,2,5), etc. If the test combination does not conflict with any of the combinations already in the combination list, it is added to the list. If all the possible combinations have been tested and no solution is found, the recursive function returns false. The algorithm is relatively efficient in terms of speed and stack memory usage. It has been implemented in C and tested with a reasonable set of test cases. For illustration, the recursive function is written in pseudocode below. When the function is called for the first time, an empty combination list and the first possible packet combination are passed as parameters.

```
function find_optimal_combination ( combination_list, test_combination )

  add test_combination to the combination_list

  if combination_list is full
    return true
  endif

  loop forever
    compute next_test_combination
    if next_test_combination does not conflict any combination in the combination_list
      if find_optimal_combination ( combination_list, next_test_combination ) == true
        return true
      else
        remove next_test_combination from the combination_list
      endif
    endif
    if next_test_combination is the last test combination
      return false
    endif
  end loop

end function
```

We have been able to find optimal frame allocation with all combinations of values for n and s shown in Table 1. However, we have not proved that the optimal frame allocation exists with all possible values of n and s. If the algorithm is not able

to find a solution, it will return false. In this case the user could try to find a suboptimal allocation, leaving some of the slots empty, or constructing packet payloads by combining smaller intermediate payloads, as explained in the following subsection. In practice, extremely long interleaving cycles are not acceptable for streaming applications, and the presented cases should be sufficient for most real-life applications.

In general, the proposed frame allocation algorithm does not conceptually differ much from conventional frame interleaving. They are both used to redistribute frame losses more smoothly. Similarly, our use of FEC provides the same average recovery rate for lost frames as the baseline generic FEC. The major difference between the proposed scheme and conventional strategies lies in the co-design of frame shuffling and FEC. Our scheme changes only the distribution of transient frame losses when different error patterns are applied to a relatively short sequence of frames.

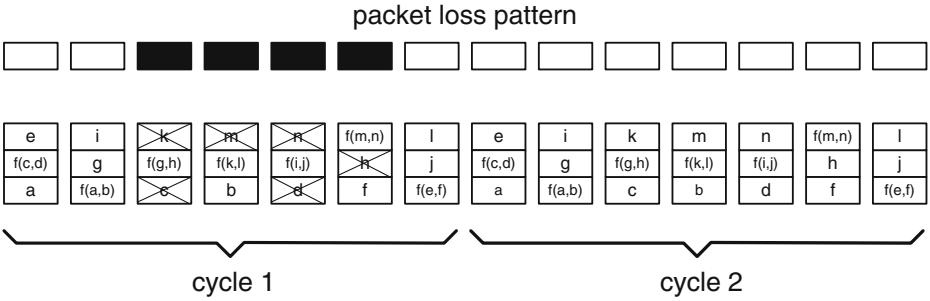
3.3 Two-step payload generation for very small frames

As the proposed packetization scheme is based on spreading consecutive media frames among a long sequence of RTP packets, an extra delay equivalent to interleaving delay is presented at both the sending end and the receiving end. As seen from Eqs. 4 and 5, the number of packets needed for one spreading cycle increases linearly and the number of frames exponentially when the number of slots in each packet is increased. This is why the proposed scheme cannot be applied as such when the media frames are very short and many frame slots are needed in each RTP payload because the interleaving latency would be far too high.

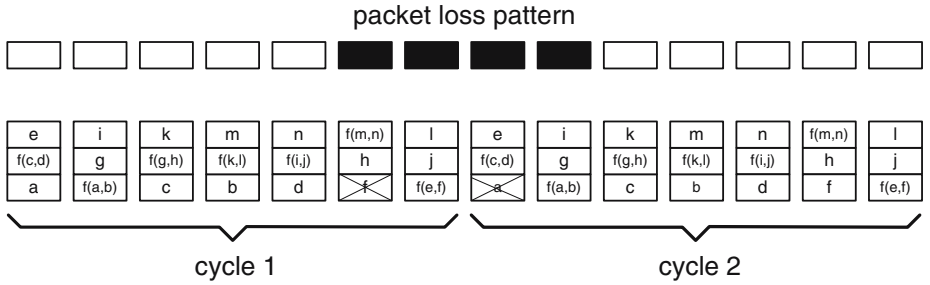
Another problem of the proposed scheme is performance instability under conditions of bursty packet losses. The impact of bursty packet losses is different if the burst hits the boundary of two different packet blocks instead of a single block. The resulting loss rate is lower if the packet losses are divided among two different cycles. This is illustrated in figure 6, where black boxes denote erased packets in the packet error pattern. Media frames unrecoverable after applying the packet loss pattern are marked with a cross. In this example, loss of four adjacent packets leads to loss of essentially different numbers of frames, depending on the position of the error burst in relation to the spread across cycles.

For short packets, it is possible to compromise between interleaving latency and header overhead by generating RTP payload in two phases. First, the frames are spread among intermediate payloads normally as explained in Section 3.2. The number of frame slots in the intermediate payloads is smaller than in the final RTP payloads, so the values for s, P and F are reasonably small. This approach also facilitates the packetization process and the calculation of the frame allocation table. In the second phase, the intermediate payloads are concatenated into final RTP packet payloads. The concatenation is performed so that the intermediate payload sequences partially overlap each other. This minimizes redundant dependencies between the resulting RTP packets and spreads bursty packet losses evenly among different cycles to avoid the problem shown in figure 6.

Figure 6 illustrates the two-step packet payload generation procedure. In this illustration, the configuration for producing the intermediate payloads is exactly the same as shown above in figure 4. In this example, the final RTP payloads have slots for six frames each, instead of the three slots in the intermediate payloads.



a) Four consecutive packet losses hitting cycle 1



b) Four consecutive packet losses divided across cycles 1 and 2

Fig. 6 Bursty packet loss (a) hits one cycle or (b) spreads across the boundary between two cycles. In the first case, six frames are lost; in the second case, only two frames are lost

It is easy to see the advantage of the two-step packetization arrangement in figure 7. Using the two-step approach, the number of frames per cycle can be limited to 42 (two intermediate payload sequences containing 21 frames each) and the number of packets to eight. Otherwise, 78 frames would be needed per cycle to fill 13 packets, as seen in Table 1. The price to pay is the increased, though still reasonable, number of redundant dependencies between packets. However, to get

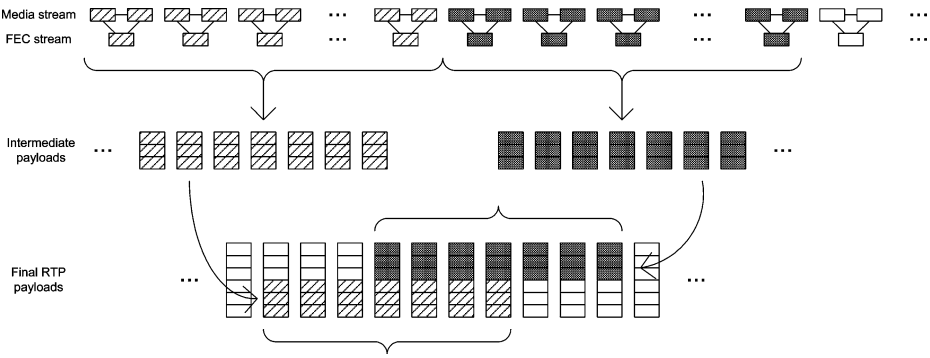


Fig. 7 Two-step packetization procedure for very short media frames

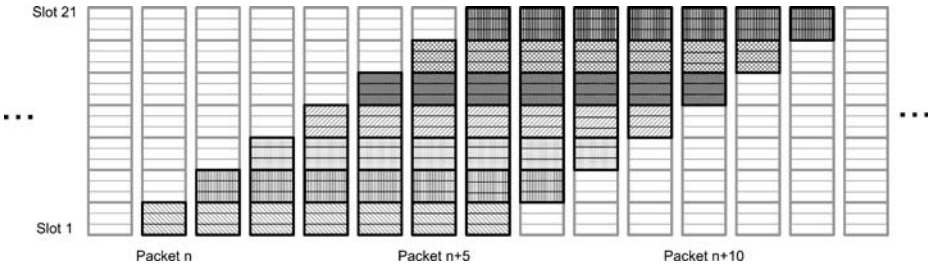


Fig. 8 Optimal alignment of small frames in packet payloads

the full advantage of this kind of frame alignment, the number of frames per packet should be larger. The ideal situation is shown in figure 8, where bursty packet losses are always positioned in the middle of some cycles but at the boundary of some others. In this example, there are 21 slots in each packet.

Typically, each audio frame represents an audio clip of a certain length. This is why interleaving delay is usually directly proportional to the number of frames in each interleaving cycle. The relative interleaving delays in the baseline FEC scheme, the proposed advanced FEC and packetization scheme and the proposed two-step packetization scheme are compared in figure 9. As the figure shows, the two-step payload generation strategy provides a good trade-off between interleaving delay and performance stability.

3.4 Verification of the proposed scheme

To verify the practical value of the proposed packetization method, we have compared it against the baseline FEC by applying different kinds of loss patterns to the packet sequences and analyzing the variance in residual frame loss rate. The sample packet configurations and the dependencies between packets were the same

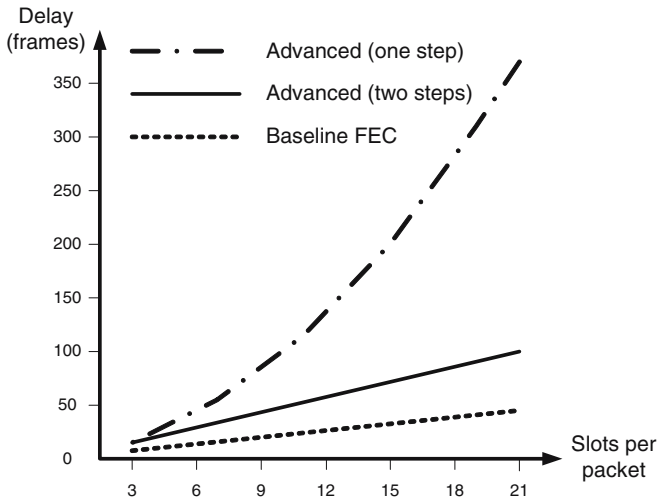


Fig. 9 Relative interleaving delays in the studied schemes compared

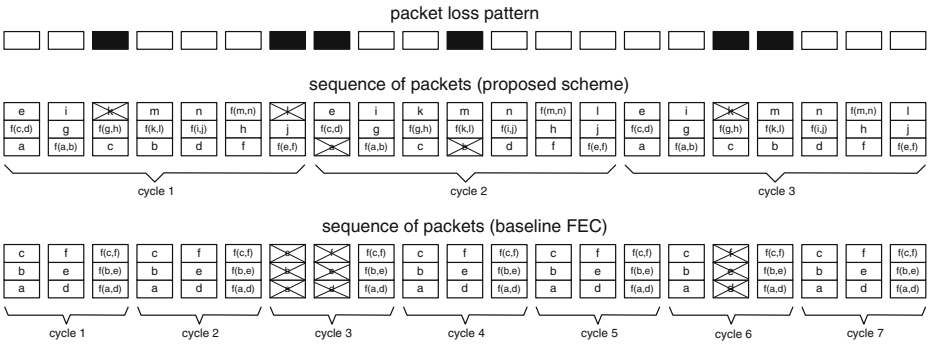


Fig. 10 Applying packet losses to the packets and analyzing the frame losses

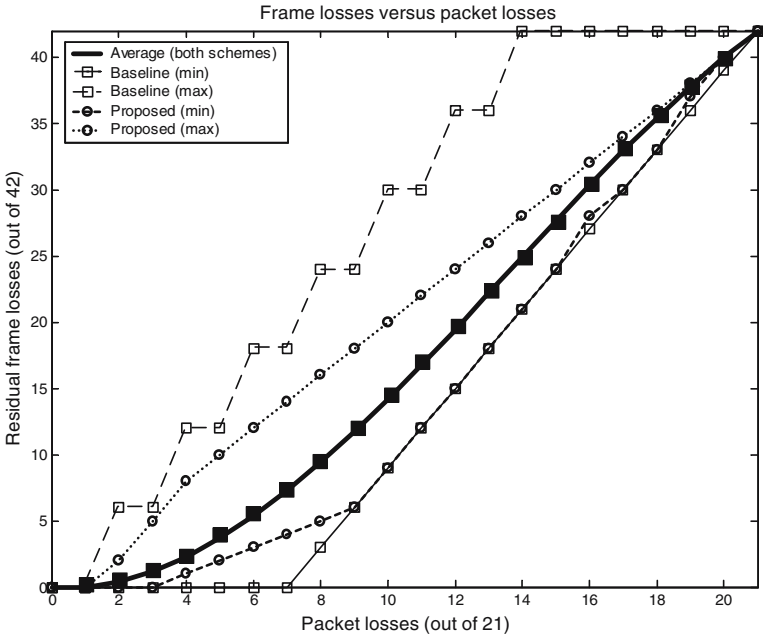
as in figures 3 and 5 above. The baseline packet sequence contained groups (cycles) of three consecutive and mutually dependent packets (two such groups in the figure) whereas in the proposed scheme, there were seven mutually dependent packets in a cycle. For the sequences to be of equal length, the test sequences comprised seven cycles in the baseline and three cycles in the proposed scheme, resulting in 21 packets in both cases. Figure 10 illustrates the test procedure, where a sample packet loss pattern is applied separately to the packets in the proposed scheme and the baseline FEC transport mode.

All the possible packet loss patterns with different numbers of packets lost were applied to both packet sequences. Table 2 shows some of the packet loss combinations, packet loss rates and average residual frame loss rates in each case. Figure 11a shows the resulting maximum and minimum bounds for the frame loss rates in the baseline and the advanced packetization scheme, respectively. Figure 11b depicts the variance of the observed residual frame loss rates.

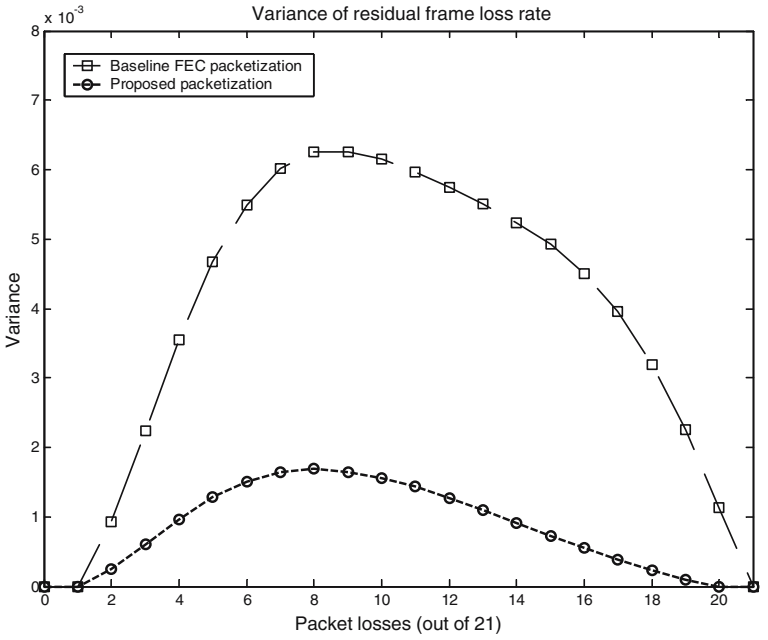
As shown in figure 11a, the average residual frame loss rate is the same in both schemes. The average frame loss rate also follows closely the theoretical values derived from Eq. 1. For simplicity in presentation, the theoretical frame loss rate is not marked in the figure. Nevertheless, it can be seen that the lower and upper bounds for the actual frame loss rate are brought closer to the average using the proposed packetization scheme. It is notable that the maximum packet loss rate always increments in steps of six frames and the minimum packet loss rate in steps

Table 2 Different packet loss pattern alternatives

Packets lost (out of 21)	Packets lost (in %)	Number of combinations	Observed residual frame loss rate (in %)
0	0	1	0
2	1	210	1
4	19	5,985	5
8	38	203,490	23
12	57	293,930	46
16	76	20,349	72
20	95	21	95
21	100	1	100



a) Residual frame loss rates



b) Variance of the frame loss rate

Fig. 11 Observed residual frame loss characteristics in the sample

of three frames when the baseline scheme is used. This is because there are three frames allocated in each packet, and in the worst case, there is always an even number of media frame packets lost. The reduction in the variance of the frame loss rate is seen even more clearly in figure 11b. These results show clearly the efficiency of the proposed packetization principle.

The example of 21 packets with three frames in each packet is a realistic scenario for many applications. For example, the frame size of high-quality audio frames encoded with MP3 or AAC is typically around 350 – 450 B, so it is possible to fit at least three of them in an IP packet with a default maximum size of 1,500 B. In the example, 21 frames would equal approximately 1 s of audio. For an average human listener, variation in transient audio quality in cycles of 1 s is clearly notable. On the other hand, quality fluctuation would be averaged out if a much longer timeframe is used.

4 Other considerations and discussion

Quality fluctuation, which causes annoyance in the user, depends on many factors such as the media type and the use scenario. Some multimedia coding schemes, especially video codecs, rely on different prediction mechanisms that inadvertently create some sort of dependence between separate frames. Therefore, loss of one frame or block may influence several other frames or blocks due to error propagation, and a small variation in frame loss rate can lead to a larger variation in subjective quality. For this reason, our proposed approach for minimizing variation in residual frame loss rate is especially useful. Our concept is useful also if smooth, high quality is especially important for the end user. A good example is music transmission (Internet radio or traditional audio streaming). In highly interactive communications, such as Internet telephony, optimization of quality plays a smaller role.

The main disadvantage of the proposed packetization scheme lies in breaking down the original order of the media frames. Because consecutive frames are spread among a long sequence of RTP packets, an extra delay equivalent to interleaving delay is presented at both the sending end and the receiving end. In addition, reordering influences the use of RTP header. Because the frames in each packet are not arranged in their original order, timestamp as defined in RTP specifications cannot be used.

It is of course possible to set the RTP timestamp according to the earliest media frame located in the payload, and use additional timestamps and sequence numbers in the payload data. However, this kind of maneuver is against the original RTP design philosophy. Indeed, such modifications undermine the nature of RTP.

Essentially, the same problem arises when conventional interleaving is used with RTP. Issues related to interleaving with RTP have been addressed in several publications, for example, by Perkins et al. [11]. Also, many RTP payload formats allow optional interleaving. The general consensus seems to be on accepting the use of interleaving with RTP. It follows then that our proposed packetization strategy should be acceptable as well.

In any case, the problem with interleaving latency remains as discussed in Section 2.1. Even though the problem in our system can be alleviated using the two-step packetization strategy described in Section 3.3, the proposed strategy is not well

suited for applications giving users seamless, constant interaction. Nevertheless, there are several applications that can benefit much from our proposed approach, including Internet radio broadcasting, audio-on-demand, and even teleconferencing with relaxed latency requirements. The basic concept is that much flexibility and notable quality improvement can be achieved even without using overwhelmingly long spreading (interleaving) cycles for packet generation.

5 Conclusions

The generic FEC schemes for RTP transmission proposed in the existing literature do not provide stable performance under varying packet loss patterns. This is because the data recovery rate in the various schemes depends not only on observed packet loss rate but also heavily on the error pattern. This is especially harmful if each RTP packet contains several independent frames from the original data stream because that means every unrecoverable packet loss typically leads to the erasure of several clustered data frames.

In this paper, we have proposed an alternative approach based on the efficient shuffling of media frames and their related FEC frames among a longer sequence of RTP packets. In terms of the residual frame loss rate under different packet loss patterns, the proposed strategy pushes the best case and worst case performance closer to the average. This significantly reduces fluctuations in the residual frame loss rate. In a typical use case, this would highly benefit the subjective quality of the resulting media output. We have verified the efficiency of our method by analyzing and comparing the variance in residual frame loss rate in two packetization scenarios, one following the baseline FEC extracted from the literature and the other following our proposed strategy.

References

1. Bormann C (ed) (2001) Robust header compression (ROHC): framework and four profiles: RTP, UDP, ESP and uncompressed. IETF RFC 3095
2. Chan K, Lu J, Chuang J (1999, May) Block shuffling and adaptive interleaving for still image transmission over Rayleigh fading channels. In: IEEE Transactions on Vehicular Technology 48(3):1002–1011
3. Claypool M, Zhu Y (2003, May) Using interleaving to ameliorate the effects of packet loss in a video stream. In: Proc. of the International Workshop on Multimedia Network Systems and Applications (MNSA). Providence, Rhode Island, pp. 508–513
4. Garey MR, Johnson DS (1979) Computers and intractability. A guide to the NP-completeness. Freeman, New York
5. 3rd Generation Partnership Project (2004) Transparent end-to-end packet switched streaming service (PSS); RTP usage model. 3GPP TR 26.937 V6.0.0
6. Korhonen J (2002, May) Error robustness scheme for perceptually coded audio based on interframe shuffling of samples. In: Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing, Orlando, Florida, pp. 2053–2056
7. Li V, Zhang Z (2002) Internet multicast routing and transport protocols. In: Proc. of the IEEE 90(3):360–391
8. Loguinov D, Radha H (2001, November) Measurement study of low bitrate internet video streaming. In: Proc. of the ACM SIGCOMM Workshop on Internet Measurement, San Francisco, pp. 281–293

9. Markopoulou A, Tobagi F, Karam M (2003, October) Assessing the quality of voice communications over internet backbones. In: IEEE/ACM Transactions of Networking 11(5): 747–760
10. Nonnenmacher J, Biersach E, Towsley J (1998, August) Parity-based loss recovery for reliable multicast transmission. In: IEEE/ACM Transactions on Networking 6(4):289–300
11. Perkins C, Crowcroft J (2000, March,) Effects of interleaving on RTP header compression. In: Proc. of the IEEE INFOCOM, Tel Aviv, Israel, pp. 111–117
12. Perkins C, Hodson O, Hardman V (1998, September/October) A survey of packet loss recovery techniques for streaming audio. In: IEEE Network 12(5):40–48
13. Rey J, Leon D, D, Rey J, Miyazaki A, Varsa V, Hakenberg R (2004, January) RTP retransmission payload format. IETF AVT Internet Draft. Work in progress
14. Rosenberg J, Schultzrinne H (1999) An RTP payload format for generic forward error correction. IETF RFC 2733
15. Schultzrinne H, Casner S, Frederick R, Jacobson V (2003) RTP: a transport protocol for real-time applications. IETF RFC 3550
16. Stockhammer T, Wiegand T, Oelbaum T, Obermeier F (2003, September) Video coding and transport layer techniques for H.264/AVC-based transmission over packet-lossy networks. In: Proceedings of IEEE International Conference on Image Processing, vol. 3, Barcelona, Spain, pp. 481–484
17. Wah BW, Su X, Lin D (2001, September) A survey of error-concealment schemes for real-time audio and video transmissions over the internet. In: Proc. of IEEE International Symposium on Multimedia Software Engineering, Taipei, Taiwan, pp. 17–24
18. Wang Y, Huang W, Korhonen J (2004, October) A framework for robust and scalable audio streaming. In: Proc. of the ACM Multimedia '04, New York, pp. 144–151
19. Zhai F, Eisenberg Y, Luna CE, Pappas TN, Berry R, Katsaggelos AK (2003, October) Packetization schemes for forward error correction in internet video streaming. In: Proc. of the Allerton Conference on Communication, Control and Computing
20. Zhu Q-F, Wang Y, Shaw L (1993, June) Coding and cell loss recovery for DCT-based packet video. In: IEEE Transactions on Circuits and Systems for Video Technology 3(3):248–258
21. Zlatokrilov H, Levy H (2004, March) Packet dispersion and quality of voice over IP applications in IP networks. In: Proc. of IEEE INFOCOM 2, Hong Kong, pp. 1170–1180

Jari Korhonen received his MSc degree in information engineering from the Department of Electrical Engineering, University of Oulu, Finland, in 2001. He joined Nokia Research Center, Tampere, Finland, as a Research Engineer in 2001. In 2004-2005 he is spending a research term at the National University of Singapore, working with Dr. Ye Wang on multimedia communications. Currently he is also pursuing towards his PhD in telecommunications at the Tampere University of Technology, Finland. His research interests include multimedia streaming, wireless real-time communications and audio coding.

Yicheng Huang received his BSc degree in computer science from the Department of Computer Science, Fudan University, China, in June 2003. He joined the National University of Singapore as a PhD candidate in September 2003. Currently, he is working with his PhD supervisor, Dr. Ye Wang, on multimedia communications. His research interests include multimedia streaming, video coding and wireless real-time communications.

Ye Wang received his Dr.-Tech. degree from the Department of Information Technology, Tampere University of Technology, Finland. In 2001, he spent a research term at the University of Cambridge, U.K., working with Prof. Brian Moore on compressed domain audio processing. He is currently an Assistant Professor with the Department of Computer Science, School of Computing, National University of Singapore.

Dr. Wang has had a nine-year career with Nokia Research Center in Finland as research engineer and senior research engineer, where he worked on Digital Audio Broadcasting (DAB) receiver prototype development, optimization of perceptual audio coding algorithms, error resilient audio content delivery to mobile phones and compressed domain audio processing for multimedia applications on small devices.

His research interests include audio compression and content-based processing, perception-aware and low-power audio processing, and error resilient content delivery to handheld devices via wireless networks. He holds a dozen patents in these areas and has published about 30 international journal and conference papers. He is a member of the technical committee, Coding of Audio Signals of the Audio Engineering Society; and a member of the Multimedia Communications Technical Committee, IEEE Communications Society.