On-the-Fly Coding for Real-Time Applications

Pierre-Ugo Tournoux, Amine Bouabdallah, Jérôme Lacan and Emmanuel Lochin
{ptournoux, amine.bouabdallah, jerome.lacan, emmanuel.lochin}@isae.fr
CNRS; LAAS
Université de Toulouse; ISAE

ABSTRACT

Although ironically it does not offer any real-time guarantee, Internet is a popular solution to support multimedia time-constrained applications (*e.g.* VoIP, Video Conferencing, ...). Following this trend, this paper focuses on the performance of these applications by studying the benefit of using a novel reliability concept which aims at significantly improving the performance of these time constrained applications over lossy best-effort networks. This reliability mechanism emerged from several recent works from both network and coding theories. Its principle is to integrate feedbacks in an on-the fly coding scheme in order to optimize the trade-off "packet decoding delay" vs "throughput". We present the first evaluations of this mechanism for VoIP and video-conferencing applications for various erasure channels. Compared to classic block-based erasure codes, the results show significant gains in terms of quality observed by the user for both applications.

Keywords

Reliability, Erasure code, VoIP, Video-conferencing

General Terms

Algorithms, Design

1. INTRODUCTION

The success of multimedia applications is well established over the Internet. However, the network does not bring any delay guarantees and an IP connection path can be composed of several link technologies that might include long-delay links. However, among these applications, some of them have strong real-time constraints. This is typically the case for video-conferencing and VoIP. For example, VoIP applications are characterized by a maximum "mouth-to-ear" threshold where a delay lower than 200 ms [2] is mandatory to get a good communication quality. As a matter of fact, the delay constraints are complex to satisfy.

This problem becomes extremely hard in the case of packet losses due to congestion or transmissions impairment. To date, two kinds of reliability mechanisms are commonly used to recover packet losses : Forward Error Correction (FEC) and Automatic Repeat reQuests (ARQ). Both are used when there is no strong delay constraints as they significantly increase the transmission delay. Indeed, retransmissions used by ARQ imply at least one additional Round Trip Time (RTT) which is unacceptable in most cases; while FEC increases the delay because of the data encoding which is done on a per-block basis. This means that the FEC decoder has to receive enough packets from a block to recover the lost data.

The encoding block size can be reduced, however for a fixed coding rate, the lower the block size, the lower the decoding performance. It means that at equal correction capability, a decrease of the block size (and consequently, of the packet decoding delay) must be followed by a decrease of the coding rate (and consequently, of the application throughput).

Some proposals to improve the relationships between *packet decoding delay*, *block length* and *throughput* are given in [8] and [6]. The combination of FEC and feedbacks, called Hybrid-ARQ, can also improve the trade-off between reliability and throughput [9], however, it does not solve the problem of the packet decoding delay, especially when transmission delays are considered.

Recently, several works on erasure and network coding proposed to include the feedbacks in the coding process in order to reduce the packet decoding delay. Sundararajan *et al.* [10] have proposed an on-the-fly network coding scheme including feedbacks. This scheme allows to reduce the average decoding delay in the case of several receivers. In their evaluation, they choose to neglect transmission delays and the delays considered result from the losses observed by different receivers. One of the main contribution of this work is the concept of "seen packet", which allows the receivers to acknowledge the degrees of freedom of the linear system corresponding to the received packets. The main advantage of this scheme is the optimization of the buffers occupancy.

Independently, Lacan and Lochin [3] have proposed another on-the-fly coding system using feedbacks in the case of point-to-point communications with high transmission delays. Basically, the principle is to add repair packets generated as a linear combination of all the source data packets sent but

not yet acknowledged. This scheme was proposed in order to enable full-reliability in delay tolerant networks and more precisely in deep space networks where an acknowledgment path might not exist or where delay might prevent the use of ARQ schemes. The analysis of this mechanism provided in [7] shows promising results.

The first contribution of our paper is the identification of these different concepts as a solution to improve real-time multimedia applications (Section 2). We choose to use the mechanism proposed in [3] and integrate the concept of "seen packets" [10] which allows to reduce slightly the mechanism's complexity. The second contribution is the analysis of the performances obtained by this mechanism with two commonly-used real-time applications over various erasure channel characteristics.

We test the performance of our proposal with a VoIP traffic (Section 3) and evaluate the results with the Mean Opinion Score value of the E-Model [2]. We show that the quality obtained is significantly higher compared to classic erasure coding schemes. Then, we evaluate this mechanism with the more stringent delay requirements of video-conferencing applications (Section 4) and show that the quality of the video obtained in terms of PSNR outperforms those observed with classic mechanisms.

2. A RELIABILITY SCHEME FOR REAL-TIME APPLICATIONS

The main principle of this erasure coding scheme is to generate repair packets from a variable-size encoding window containing the source packets P_i, $i > 0$, provided by the source application and not yet acknowledged by the receiver. We denote $R_{(i..j)} = \sum_{u=i}^{j} \alpha_u^{(i,j)}.P_u$ a repair packet built as a linear combination of source packets ranging from i to j. The coefficients $\alpha_u^{(i,j)}$ are randomly chosen in the finite field $\mathbb{F}_q$ and are transmitted in the header of the repair packet. A repair packet is generated each k source packets, where k, which determines the coding rate (equal to $1/(k+1)$) is determined according to the packet loss rate. The source packets are sent unmodified (systematic code).

At the receiver side, the decoder "subtracts" each received source packet to all received repair packets in which it is involved. It follows that the set of resulting repair packets is a set of linear combinations of the lost source packets. The decoding consists simply in solving this linear system. This can be done as soon as the number of received repair packets (the equations) is at least equal to the number of lost source packets (the unknowns).

The encoding window is emptied by the feedbacks of the decoder which acknowledge the reception or the decoding of the source packets. The feedback contains a packet sequence number below which all the previous packets are known. The packets with a sequence number higher to this number are included in a selective acknowledgment vector. This acknowledgment scheme allows to reduce the number of source packets involved in the encoding/decoding at a given time while it maintains the full-reliability property.

In order to illustrate the properties and the overall behavior of this mechanism, we give an example of a simple data exchange in Fig. 1. In this example, during the data exchange, packet P_2 is lost. However, the repair packet $R_{(1,2)}$, successfully arrived, allows to rebuild P_2. We assume that the receiver sends an acknowledgment packet to inform the sender that it can compute the next repair packets from packet P_3 (*i.e.* this packet acknowledges packets P_1 and P_2). Then, this acknowledgment is lost. However this loss does not compromise the following transmissions and the sender simply continues to compute repair packets from P_1. After this, we see that P_3, P_4 and $R_{(1..4)}$ packets are lost. None of these packets need to be retransmitted as they are rebuilt thanks to the packets received from P_5 to $R_{(1..8)}$. Indeed, the receiver can rebuild P_3, P_4 by firstly "subtracting" all the received source packets from the repair packets in order to obtain $(R'_{(1..6)}, R'_{(1..8)})$ which are linear combinations of P_3 and P_4. By solving this linear system, P_3 and P_4 can be recovered.

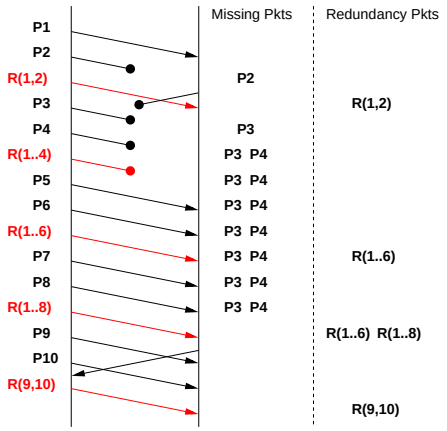
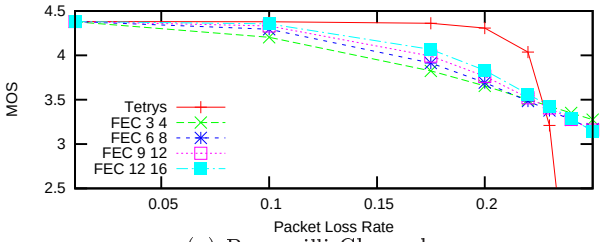


Figure 1: A simple data exchange

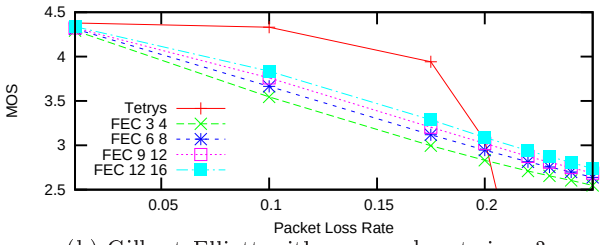
In this reliability scheme the purpose of the acknowledgments is to reduce the number of source packets handled by a repair packet as illustrated in Fig. 1, where, after receiving the acknowledgment, the repair packet $R_{(9,10)}$ starts from 9 and not from 1. The seen packet concept introduced in [10] allows to remove packets from the encoding window even if they are not yet decoded. As an example after the reception of $R(1..6)$ the receiver was allowed to acknowledge P_3 as a seen packet, reducing slightly the size of the encoding window at the sender side. The receiver was also able to acknowledge P_5 and P_6 with a SACK block reducing more significantly the size of the window. Finally, we can notice that the loss of acknowledgments does not compromise reliability (in Fig. 1, if no feedback is received, the sender simply sends $R_{(1..10)}$), even if it increases the coding window and consequently it impacts complexity.

To summarize this short introduction, this scheme permits a full-reliability even if some source data packets, repair packets or acknowledgments are lost. More interestingly for the context of real-time applications, the decoding does not depend on the feedbacks and thus the loss recovery delay is totally independent from the RTT.

3. VOIP WITH TETRYS



(a) Bernouilli Channel



(b) Gilbert Elliott with average burst size=3

Figure 2: MOS for the G711 codec as a function of the PLR. The encoding ratio is fixed to 3/4 and the FEC performances are shown with different block size.

This section presents an evaluation of the performance of Tetrys for the VoIP application. Since the delays due to retransmissions are in most cases unacceptable, we do not consider ARQ and we compare Tetrys with block-based FEC. We show a subset of the simulation settings that we tested with notably various Packet Loss Rate (PLR) generated with Bernoulli (i.i.d. losses) 2(a) and Gilbert-Elliott (bursty losses) 2(b) channels. Our results are issued from a standalone simulator which models the link layer losses events and the protocol's behavior. Since the block size impacts on the FEC performances for a given coding rate, we also show the result of FEC for a coding rate of 3/4 and for block of sizes 4, 8, 12 and 16. To obtain the same coding rate, Tetrys generates a repair packet every 3 source packets. The repair packets (both FEC and Tetrys) are sent as soon as possible (depending on the link layer capabilities) after their computation. The transmission delay is fixed to 100ms and a source packet is generated every 10ms. At the receiver side, we measure the observed packet loss rate and the delay (including the transmission and the decoding delays). These two values are used to estimate the Mean Opinion Score (MOS) obtained thanks to the E-model [2] as a quality metrics.

Figures 2(a) and 2(b) show the MOS factor evolution as a function of the PLR for the G711 codec. The two subfigures differ from the type of underlying channel; Fig 2(a) and 2(b) show the MOS for respectively a Bernoulli channel and a Gilbert Elliott channel with an average burst size of 3. On the Bernoulli channel, we can see that for a 20% packet loss rate, Tetrys obtains a stable and significantly better MOS value compared to FEC mechanisms. Tetrys rebuilds the lost packet under acceptable delay while the packets not rebuilt by FEC impact its MOS factor. This comparison of the mechanisms remains true for the bursty channel. One can also notice that user's experience remains "good" (MOS $\geq$ 4.0) for FEC up to PLR equal to 12.5% and 5% for respectively Bernoulli and Gilbert-Elliott channels, whereas Tetrys obtains "good" quality for PLR up to respectively 17.5% and 14%. A given set of parameters is then usable under a larger breathing space than FEC. It also permits to increase the coding rate, allowing then to reduce the load of the network due to the repair packets.

We can see that when the MOS for Tetrys becomes smaller than the MOS of FEC, the quality of FEC is between 3.4 and 2.8 which is considered as "poor" by the users. This means that when the performance of Tetrys decreases, the one of FEC is already not acceptable.

4. VIDEO-CONFERENCING WITH TETRYS

We choose to present the performance of Tetrys in a time-constrained and dynamic context. Indeed, amongst real-time applications, video-conferencing requires significantly less than one second end-to-end delay and is known to perform ideally when the delay does not exceed 100 ms (see [11]). Video application is also characterized by its natural variable bit rate (VBR), even if recent video coders (as H.264) can generate constant bit rate (CBR) on average. With regards to Tetrys, these two last characteristics of the video conference applications impose, according to the instantaneous video bit-rate, a variable upper limit on the usable (not obsolete) number of packets at the receiver side. Consequently, at the sender side, it indirectly defines a variable upper limit on the coding window size, (beyond that upper limit, the coded packets would be late anyway at the receiver and will not be considered). While FEC-based scheme cannot benefit much from this property since it operates on fix-block length whatever the upper limit on the usable packets on the receiver side, Tetrys automatically expands its coding window when data losses occur, hence improving the probability of recovering from losses. Since, the higher the source bit rate, the bigger the usable coding window (for a fix end-to-end delay), Tetrys hence provides a stronger protection to packets that belong to high bit rate data. For video coded data, since I-frames have the higher bit-rate, they will benefit most from the adaptability of Tetrys. Furthermore, it is also known that I-frames, when lost, can have a worse impact on the end user quality compared to P-frames or B-frames. It follows that, in addition to its excellent performance on constant-bit rate flows, Tetrys intrinsically provides a kind of unequal error protection like other (less flexible) erasure schemes specifically designed [1]. Consequently, we expect significant improvement of the video quality in conversational video application when using Tetrys compared to classic FEC schemes.

Contrary to error resilient coding scheme like [4], Tetrys aims to be used at the transport layer level and it is not specific to any codec or application. That is the reason why, as for VoIP, we compare Tetrys with variable FEC schemes under variable loss rate in Bernoulli and Gilbert-Elliott channels. As a video codec, we use the last ITU-T's video codec Recommendation H.264 and the JM 15.1 H.264/AVC software. We ran our simulation on Foreman sequence, in CIF size, with a frame skip of one picture resulting in a frame rate of 15 fps. One I-frame is inserted every 14 P-frames and no B-frames were inserted because of the extra delay they generate. The average bit-rate is about 384 kbps at the out-

Channel Type	PLR	Tetrys	FEC			
			3 4	6 8	9 12	12 16
Bernouilli	5%	35.67	33.47	34.6	32.32	30.91
	10%	34.38	26.75	31.39	29.19	27.46
	15%	31.77	20.26	24.69	25.58	23.34
	20%	26.02	15.32	18.70	20.05	20.23
Gilbert Eliot with average burst size = 3	5%	32.66	27.48	30.17	30.85	31.30
	10%	28.75	20.88	24.05	24.89	25.86
	15%	24.11	16.70	19.48	20.85	21.39
	20%	18.81	15.03	16.42	17.87	18.12

Table 1: PSNR of FEC and Tetrys

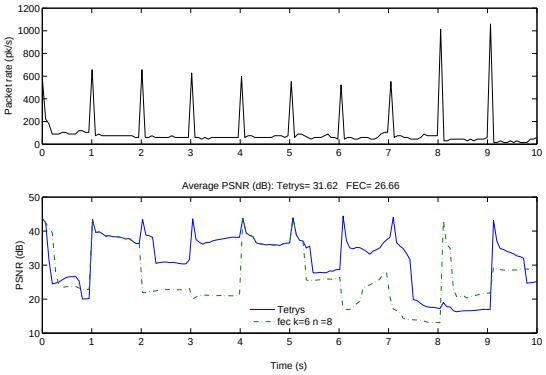


Figure 3: Packet rate and instantaneous PSNR of a live video with PLR of 15% on the Bernoulli channel

put of the video coder and the coded stream is packed into packets of 500 bytes length. The maximal tolerable end-to-end delay is set to 200 ms, hence all packets received after this due time are dropped. A total of 150 coded frames corresponding *to* 10 s of video is used. In order to obtain representative results, each test is repeated 20 times (corresponding to 3000 frames and 200 s of video). This setup is derived from the common testing conditions mentioned in [11]. For evaluating the video we use the Evalvid framework described in [5] where the video quality is measured with the *Peak Signal to Noise Ratio* (PSNR) metric where high scores means better quality.

As illustrated through experiments given in Table 1, Tetrys clearly outperforms all the tested FEC schemes in all scenarii. For the average performances on the Bernoulli channel, Tetrys achieves a gain of 6.19 dB at a loss rate of 15 % over the best FEC scheme (FEC 9 12) for this scenario. Furthermore, the drop in PSNR for Tetrys does not exceed 4 dB when the packet loss rate increases from 5% to 15% ; hence ensuring the average PSNR is always above 30 dB. When full reliability is impossible because of high time-constraints, Tetrys allows graceful degradation of the video quality. On the Gilbert-Elliott Channel, the results keep the same tendency even if the gains are less important when the burst length increases (2.72 dB for average burst length of 3 and FEC 12 16). These significant gains are achieved thanks to the flexibility of Tetrys. This particularity of Tetrys is better illustrated in Figure 3 whose bottom graph gives a snapshot of 10 seconds of the evolution of the instantaneous PSNR and the top graph represents the instantaneous packet rate of the video data where peak points represent the occurrence of I-frames. We can clearly see on this figures that Tetrys retrieves 9 I-frames out of 10, whereas FEC schemes succeeded 5 times. Statistics processed on the whole sequence showed that Tetrys retrieves on average 84% I-frames whereas FEC schemes succeeded only 60% of the time over a Bernoulli channel and respectively 73% and 40% over a Gilbert Elliott channel.

And last but not least, for video data, it happened that the high bit rate data has the most important effect on the end user video quality. Consequently, it turns out that Tetrys is providing efficient transparent unequal protection to video data. It is worth noting that Tetrys does not require any extra information exchanges (about data types and sizes and so forth) from the application while most of the existing unequal protection schemes do.

5. CONCLUSION

These results are encouraging and give a proof of the viability of the on-the-fly coding for real-time application. We gave two examples of real-time application to which full-reliability is not mandatory. Future work will include the extension of these first performance evaluations on various kinds of applications. The analysis of the behavior of the mechanism, notably the choice of the redundancy ratio according to the context and application requirements will be deeply studied. The simulations showed that the coding/decoding complexities are low compared to block codes, but this must be confirmed by theoretical analysis.

6. REFERENCES

[1] A. Bouabdallah and J. Lacan. Dependancy-aware unequal erasure protection codes. PV 15^{th} Int. Packet Video Workshop, 7:27–33, May 2006.

[2] ITU. T-REC-G.107-200503-I, The E-Model, a computational model for use in transmission planning.

[3] J.Lacan and E. Lochin. Rethinking reliability for long-delay networks. In *Int. Workshop on Satellite and Space Com., (IWSSC)*, pages 90–94, Oct. 2008.

[4] M. Kim, H. Oh, N. Dutt, A. Nicolau, and N. Venkatasubramanian. Pbpair: an energy-efficient error-resilient encoding using probability based power aware intra refresh. *SIGMOBILE Mob. Comput. Commun. Rev.*, 10(3):58–69, 2006.

[5] J. Klaue, B. Rathke, and A. Wolisz. Evalvid - A framework for video transmission and quality evaluation. In *Performance Tools*, pages 255–272, 2003.

[6] J. Korhonen and P. Frossard. Flexible forward error correction codes with application to partial media data recovery. *Signal Processing : Image Communication*, 24:229–242, 2009.

[7] J. Lacan and E. Lochin. On-the-fly coding to enable full reliability without retransmission. *http://arxiv.org/abs/0809.4576*, 2008.

[8] E. Martinian and C.-E. W. Sundberg. Burst erasure correction codes with low decoding delay. *IEEE Transactions on Information Theory*, Oct. 2004.

[9] A. Sahai. Why do block length and delay behave differently if feedback is present? *IEEE Transactions on Information Theory*, 54(5):1860–1886, May 2008.

[10] J. K. Sundararajan, D. Shah, and M. Medard. ARQ for network coding. *IEEE Int. Symp. on Information Theory*, pages 1651–1655, July 2008.

[11] S. Wenger. H.264/AVC over IP. *IEEE Trans. Circuits Syst. Video Techn*, 13(7):645–656, 2003.