Erasure Coding and Congestion Control for Interactive Real-Time Communication

Pierre-Ugo Tournoux[2], Tuan Tran Thai[1], Emmanuel Lochin[1], Jérôme Lacan[1], Vincent Roca[1]

[1]Université de Toulouse, France   [2]NICTA, Sydney, Australia   [3]INRIA, Grenoble, France

I. PROBLEM STATEMENT

The use of real-time applications over the Internet is a challenging problem that the QoS epoch attempted to solve by proposing the DiffServ architecture. Today, the only existing service provided by the Internet is still best-effort. As a result, multimedia applications often perform on top of a transport layer that provides a variable sending rate. In an obvious manner, this variable sending rate is an issue for these applications with strong delay constraint. In a real-time context where retransmission can not be used to ensure reliability, video quality suffers from any packet losses. To illustrate this problem and motivate why we want to bring out a certain class of erasure coding scheme inside a multimedia congestion control protocol such as TFRC [1], we propose this simple simulation scenario: we encode a CIF 'Foreman' sequence of 300 frames with two values of Quantization Parameter (QP). For a Group of Picture (GOP) size of 30 frames, QP 28 and 29 result in H.264/AVC videos of 531.13 kb/s (Peak Signal to Noise Ratio (PSNR) of 36.9 dB) and 441.37 kb/s (PSNR of 36.2 dBs), respectively. A video with QP of 28 generates 16.9 % of redundancy bit rate compared to the one of 29. We drive the simulations with encoded video of QP 29 protected by an erasure coding scheme (Tetrys [11]) with redundancy of 12.5% and the one of 28 without protection. We randomly generate a loss rate of 1 %. In Fig. 1, we observe a strong quality degradation of the video without protection (blue line compared to purple line) while Tetrys recovers all losses within one way End-to-End delay constraint of 150 ms (red and green lines are identical). This implies that a slightly higher quality without protection suffers from any losses while a slightly lower quality video with protection can sustain up to a certain loss rate (3% in these simulations) depending on its redundancy. As a matter of fact, erasure-coding seems mandatory to enable a fair real-time video quality.
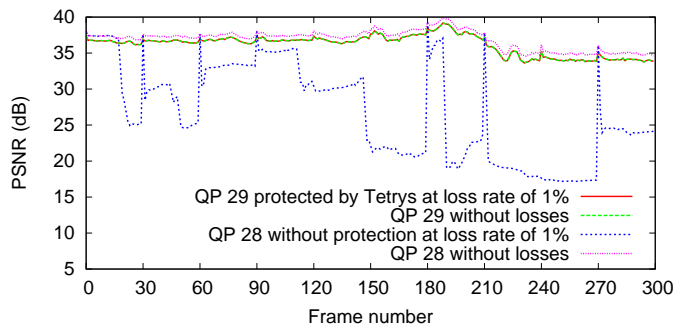

Fig. 1. Video quality variation in presence of packet losses with and without erasure coding

II. HOW TO COPE WITH THESE LOSSES?

Currently, there are two kinds of reliability mechanisms based respectively on retransmission and redundancy schemes. Automatic Repeat reQuest (ARQ) schemes recover all lost packets thanks to retransmissions. This implies that the recovery delay of a lost packet needs at least to wait one supplementary Round Trip Time (RTT) and this supplementary delay might exceed the threshold above a real-time application considers a packet outdated.

A well-known solution to prevent this additional delay is to add redundancy packets to the data flow. This can be done with the use of Application Level Forward Error Correction (AL-FEC) codes. AL-FEC codes are FEC codes for the erasure channel where symbols (i.e., packets) are either received without any error or lost (i.e., erased) during

transmission. The addition of $n-k$ repair packets to a block of k source packets allows to rebuild all of the k source packets if a maximum of $n-k$ packets are lost among the n packets sent. In practice, only Maximum-Distance Separable codes (MDS), such as Reed-Solomon codes [4], have this optimal property, whereas other families of codes, such as LDPC [7], need to receive a few number of extra symbols in addition to the k strict minimum. However, if more than $(n-k)$ losses occur within a block, decoding becomes impossible. In order to increase robustness (e.g., to tolerate longer bursts of losses), the sender can choose to increase the block size with the price of an increase of the decoding delay or to add more redundancy while keeping the same block size with the price of a decrease of the goodput. These trade-off between *packet decoding delay; block length* and *throughput* are, for instance, addressed in [2].

Recently, a novel coding approach has been proposed [3], [5], [11]. We propose here Tetrys, a systematic erasure code, inside a congestion control protocol totally departs from the above schemes. Unlike current reliability methods, this on-the-fly coding scheme allows to fill the gap between systems without retransmission and fully reliable systems by means of retransmissions. Configuring Tetrys for full reliability requires only the knowledge of the average PLR. Indeed, Tetrys needs only an redundancy ratio to perform as its configuration does not depend of the size of the burst of losses. The main properties of Tetrys is (1) to be tolerant to any burst of source, repair or acknowledgement losses, as long as the amount of redundancy exceeds the packet loss rate (PLR), and (2) the lost packets are recovered within a delay that does not depend on the RTT, which is a key property for real-time applications. We believe that these properties make Tetrys a good reliability candidate for a congestion control protocol such as TFRC [1].

III. ISSUES FOR THE CONGESTION CONTROL

The use of erasure codes as a reliability mechanism requires to identify the key properties of the congestion control that they have to be paired with. The next two sections detail the relevant properties of the current congestion control and then describes the potential solution.

A. *Congestion control and real-time variable bit rate*

TCP have been long know as an unsatisfying solution for carrying real-time traffic. Indeed, the head of line blocking problem induce a significant delay in the communication of application that tolerate few losses. Sending a packet requires to wait for an acknowledgement which, in case of losses or slow-start, tends to arrive by burst hence increasing the buffering delay at the sender side. Finally, despite the fact that TCP's AIMD reaches a steady-state, its saw-tooth behavior prevents the application to adapt efficiently its bit rate efficiently. Furthermore, the buffering at the sender side might overtake the delay constraint of the application. As a result, TCP would support real-time traffic if the fair-share is at least twice bigger than the source bit rate [12]. For all these reasons, the support of real-time applications has turned towards protocols allowing out-of-order delivery and rate-based congestion control.

TFRC [1] is a congestion control mechanism designed to allow applications that use fixed packet size to compete fairly with TCP flows using the same packet size *e.g.* 1500 bytes. If some real-time applications such as VoIP find a satisfying solution in TFRC, video-conferencing -which is characterized by a variable bit rate and a variable packet size- experiences severe performance issues when their sending rate is controlled by TFRC. As TFRC acts as a token bucket, the burst of packets has to be queued at the sender side before it can be entirely sent, thus impairing the interactivity and inducing losses in case of stringent delay constraint. The usual way to counter this drawback is to use padding and constantly transmit at the burst rate *e.g.* I-frames' packet rate in case of video. Obviously, it requires the fair-share to be much bigger than the application source rate and it reduces the overall network goodput. Another drawback of TFRC is the packet size that is assumed to be fixed and similar to the packet size of the concurrent TCP flows. Smaller packets (such as the one generated by P-Frames) are counted as full size packet and the required fair-share is much bigger than the actual rate generated by the codecs.

Few work have been proposed enable VBR sources in TFRC. In [9], the authors highlight the mismatch between the media rate and TFRC sending rate and propose to combine Faster Restart to TFRC to better support bursty applications. Although their study shows a certain improvement in terms of quality of experience for the user, the gain obtained is limited to application that follows a on-off process. The issue induced by the packet size is also partially tackled in TFRC-VP [8] which proposes to send small fixed size packets while being fair with TCP flows sending MTU sized packets.

B. Coupling congestion control, real-time application and erasure coding

To the best of our knowledge, there is no transport protocol allowing source burstiness and variable packet size and hence they don't allow to fully exploit the improved compression and interactivity allowed by modern codecs. While remaining TCP-friendly, the ideal solution should not to delay the sending of the data and the packet size should be arbitrary small. As a side effect, it would significantly ease the dimensioning of the erasure code in charge of loss recovery. We are aware that such a freedom in the packet timing might be hard to achieve -if not orthogonal to TCP-friendliness- and hence it should be bounded. Towards this end, the equation based congestion control TFMCC-VP [13] allows to trade packet size for packet rate while remaining fair with TCP. With TFRC, when congestion occurs, the packet rate reduction induce either a higher decoding delay of more losses unrecovered by the erasure codes. To perform correctly, the later will need a redundancy ratio, thus leaving less rooms to the application's data. Instead of adding a new unknown parameters that will impair the correction capability of the erasure code, TFMCC-VP [13] allows the packet rate can remain the same. As emphasized in [11], on-the-fly erasure codes have a strong advantage compared to erasure block codes as for a given redundancy ratio, they shows the same performance than the best configuration among the erasure block codes. We consider that the robust configuration of on-the-fly codes combined with the fixed (or configurable) packet rate of TMFCC-VP allows a promising solution to enable congestion control for interactive real-time communication.

In the context of video streaming with loose delay constraints, the authors of [10] follow a strongly different approach as they consider that the redundancy should not be part of the congestion-controlled rate. Their erasure block code configuration scheme predicts the loss rate and sends the minimum coding rate sent in addition to TFRC's rate. However their scheme does not tackle the case of applications with strong delay constraints. The proposal of [6] is pushing further this idea and suggests that the redundancy packets should be marked and discarded following an AQM drop precedence policy. Despite the deployment of such scheme is questionable, we believe that solutions that send redundancy packets outside the congestion-controlled flow also deserve to be investigated.

REFERENCES

[1] S. Floyd, M. Handley, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification, September 2008.

[2] Jari Korhonen and Pascal Frossard. Flexible forward error correction codes with application to partial media data recovery. *Signal Processing: Image Communication*, 24, 2009.

[3] J. Kumar Sundararajan, D. Shah, and M. Medard. ARQ for network coding. In *IEEE International Symposium on Information Theory*, pages 1651 –1655, july 2008.

[4] J. Lacan, V. Roca, J. Peltotalo, and S. Peltotalo. Reed-Solomon Forward Error Correction (FEC) Schemes. RFC 5510 (Proposed Standard), April 2009.

[5] Jerome Lacan and Emmanuel Lochin. Rethinking reliability for long-delay networks. In *Proceedings of International Workshop on Satellite and Space Communications IWSSC*, Oct. 2008.

[6] Ratul Mahajan, Jitendra Padhye, Ramya Raghavendra, and Brian Zill. Eat all you can in an all-you-can-eat buffet: A case for aggressive resource usage, 2008.

[7] V. Roca, C. Neumann, and D. Furodet. Low Density Parity Check (LDPC) Staircase and Triangle Forward Error Correction (FEC) Schemes. RFC 5170 (Proposed Standard), June 2008.

[8] Floyd S. and Kohler E. TCP Friendly Rate Control (TFRC): The Small-Packet (SP) Variant. RFC 4828 (Experimental), April 2007.

[9] Arjuna Sathiaseelan and Gorry Fairhurst. TCP-Friendly Rate Control (TFRC) for bursty media flows. *Computer Communications*, 34(15):1836–1847, 2011.

[10] H. Seferoglu, A. Markopoulou, U. C. Kozat, M. R. Civanlar, and J. Kempf. Dynamic fec algorithms for tfrc flows. *Trans. Multi.*, 12(8):869–885, December 2010.

[11] Pierre-Ugo Tournoux, Emmanuel Lochin, Jérôme Lacan, Amine Bouabdallah, and Vincent Roca. On-the-fly erasure coding for real-time video applications. *IEEE Transactions on Multimedia*, 13(4):797–812, 2011.

[12] Bing Wang, Jim Kurose, Prashant Shenoy, and Don Towsley. Multimedia streaming via TCP: An analytic performance study. *ACM Trans. Multimedia Comput. Commun. Appl.*, 4(2):16:1–16:22, May 2008.

[13] Jörg Widmer, Catherine Boutremans, and Jean-Yves Le Boudec. End-to-end congestion control for TCP-friendly flows with variable packet size. *SIGCOMM Comput. Commun. Rev.*, 34(2):137–151, April 2004.