To link to this article: DOI: 10.1016/j.image.2014.02.003
URL: http://dx.doi.org/10.1016/j.image.2014.02.003

To cite this version: Tran-Thai, Tuan and Lacan, Jérôme and Lochin, Emmanuel *Joint on-the-fly network coding/video quality adaptation for real-time delivery*. (2014) Signal Processing: Image Communication, vol. 29 (n° 4). pp. 449-461. ISSN 0923-5965

Joint On-the-Fly Network Coding/Video Quality Adaptation for Real-Time Delivery

Tuan Tran Thai[a], Jérôme Lacan[a], Emmanuel Lochin[a]

[a] *University of Toulouse; ISAE/DMIA; TéSA; Toulouse, France*

Abstract

This paper introduces a redundancy adaptation algorithm based on an on-the-fly erasure network coding scheme named Tetrys in the context of real-time video transmission. The algorithm exploits the relationship between the redundancy ratio used by Tetrys and the gain or loss in encoding bit rate from changing a video quality parameter called the Quantization Parameter (QP). Our evaluations show that with equal or less bandwidth occupation, the video protected by Tetrys with redundancy adaptation algorithm obtains a PSNR gain up to or more than 4 dB compared to the video without Tetrys protection. We demonstrate that the Tetrys redundancy adaptation algorithm performs well with the variations of both loss pattern and delay induced by the networks. We also show that Tetrys with the redundancy adaptation algorithm outperforms traditional block-based FEC codes with and without redundancy adaptation.

Keywords: real-time transmission, network coding, quality adaptation

1. Introduction

Video traffic currently plays an important role on the Internet. The delivery of multimedia content has been extensively studied to provide better service and quality to end users. H.264/AVC (Advanced Video Coding), a video coding standardized since 2003, has shown better compression performance than previous standard codecs such as MPEG-4 Part 2, H.263 [1]. Additionally, the newly standardized video codec, High Efficiency Video Coding (HEVC) [2] provides up to 50% bit rate savings for equivalent perceptual quality compared to H.264/AVC. However, the higher compression efficiency makes the encoded video more sensitive to errors and losses during transmission on networks. A small number of losses can significantly degrade the video quality perceived by end users. Thus, the challenge in real-time video transmission over error prone networks is twofold:

1. Video traffic must be protected from losses over the Internet. Indeed, Wenger showed that the Peak Signal to Noise Ratio (PSNR) decreases up to several dB when the loss rate is greater than 1% [3]. From the video perspective, error resilience tools [3, 4] (e.g., data partition, Flexible Macroblock Ordering) provided by the video codec standards are designed to mitigate the impact of packet loss. However, these tools usually use extra bit rate, which leads to lower coding efficiency [5]. From the network perspective, the obvious way to provide reliability is retransmission as TCP does. Nevertheless, the delay to recover the lost packets requires at least one additional Round Trip Time (RTT) which is suitable for interactive applications. The traditional approach is to use Forward Error Correction (FEC) [6] to protect the video from losses. The main problem of this block code scheme is that it requires dynamically adapting its initial parameters and as a result, complex probing and network feedback analysis. Recently, novel erasure network coding approaches that prevent such complex configuration have been proposed [7, 8, 9]. The main difference between these proposals is that the code in [8, 9], called Tetrys, is more suitable for real-time video applications as this code is systematic and the repair packets in [8, 9] are equally distributed between data packets.

2. The network condition (e.g., delay, loss rate) varies over time. Hence, it requires an enhanced mechanism for erasure codes to adapt to network dynamics. In [10], the authors propose a Random Early Detection FEC mechanism in the context of video transmission over wireless networks. This mechanism adds more redundancy packets if the queue at the Access Point is less occupied and vice versa. However, this approach assumes that the wired segment of the network is loss free. In reality, the wired segment of the network might experience packet losses due to cross traffic or network congestion. The approach in [11] switches between different FEC techniques to adapt to the state of the network in the context of multi-source streaming.

Sahai in [12] showed the more the redundancy introduced on the network the shorter the packet recovery delay. Tetrys exhibits the same behavior for the stationary channel [9]. However, when the channel state varies over time, it is more difficult to control the variation of the redundancy ratio. Thus, in this article, we propose a redundancy adaptation algorithm based on Tetrys that we call A-Tetrys or Adaptive Tetrys to cope with network dynamics (e.g., loss rate and delay variations) in the context of real-time video transmission. Our algorithm adapts the Tetrys redundancy ratio by increasing or decreasing the video quality to deliver video in which the residual packet loss rate is minimized as much as possible within the delay constraint required by the application. Furthermore, we choose the Tetrys redundancy ratio list so that the video with slightly lower quality protected by Tetrys does not send more bit rate than the video with higher quality but without protection from the erasure codes. The results show that A-Tetrys gains on average up to or more than 1 dB compared to standard Tetrys and more than 4 dB compared to the video without protection. The standard Tetrys is referred to as the original Tetrys where the redundancy ratio is fixed during transmission. The simulation results show that A-Tetrys adapts well to both loss pattern and delay induced by networks. We also show that A-Tetrys outperforms FEC with and without redundancy adaptation algorithm.

The rest of this article is organized as follows: Section 2 briefly introduces the principle of Tetrys and notes some important properties. The redundancy adaptation algorithm based on Tetrys is described in detail in Section 3. Section 4 presents the rationale behind a chosen redundancy list for H.264/AVC real-time transmission. Section 5 studies the impacts of algorithm parameters using Constant Bit Rate traffic. The evaluation with video traffic is presented in section 6. Tetrys compared with FEC is the topic of Section 7. Section 8 discusses the differences between our approach and existing work. Concluding remarks are given in Section 9.

2. Tetrys overview

Tetrys [9] is an erasure network coding scheme that uses an elastic encoding window buffer B_{EW}. This buffer stores all source packets transmitted but not yet acknowledged. For every k source packets, the Tetrys sender sends a repair packet $R_{(i..j)}$ which is built as a linear combination of all packets currently in B_{EW} from packets indexed i to j

$$R_{(i..j)} = \sum_{l=i}^{j} \alpha_l^{(i,j)}.P_l$$

where the coefficients $\alpha_l^{(i,j)}$ are randomly chosen in the finite field $\mathbb{F}_q$. Through this coding, the redundancy ratio is specified by $1/(k+1)$ or $1/n$ (where $n = k+1$) which corresponds to a code rate $k/(k+1)$. Unlike TCP that acknowledges every received packet, the Tetrys receiver is only expected to periodically acknowledge the received or decoded packets. Upon reception of an acknowledgment packet, the Tetrys sender removes the acknowledged source packets out of its B_{EW}. Generally, the Tetrys receiver can decode all lost packets as soon as the number of received repair packets is equal to the number of lost packets. By this principle, Tetrys is tolerant to burstiness losses in both source, repair and acknowledgment packets as long as the redundancy ratio exceeds the packet loss rate (PLR). Furthermore, the lost packets are recovered within a delay that does not depend on the Round Trip Time (RTT). This property is very important for real-time applications where the time constraint is stringent.

Fig. 1 shows a simple Tetrys data exchange with $k = 2$ which implies that a repair packet is sent for every two sent source packets (i.e., a redundancy ratio of 33.3%). The packet P_2 is lost during the data exchange. However, the reception of repair packet $R_{(1,2)}$ allows the reconstruction of P_2. When the acknowledgment event occurs, the Tetrys receiver sends a Tetrys acknowledgment packet that acknowledges packets P_1 and P_2. However, if this acknowledgment packet is lost, this loss does not interrupt the transmission; the sender simply continues to build the repair packets from P_1. Later, the lost packets P_3, P_4 are reconstructed thanks to $R_{(1..6)}$ and $R_{(1..8)}$. It must be noted that the reception of packet $R_{(1..6)}$ does not allow the recovery of the first lost packet observed (packet P_3) since last packet recovery event (the reception of packet $R_{(1,2)}$). Indeed, the packet P_4 is still missing from the linear combination in packet $R_{(1..6)}$. The reception of a second acknowledgment packet allows the sender to remove the acknowledged source packets and build the repair packets from P_9. Further details can be found in [9].

This example makes two important points. First, all lost packets (the first lost packet since the last packet recovery event as well as the last lost packet observed) are recovered altogether. Indeed, the Tetrys receiver has to wait until the number of repair packets is equal to the number of lost packets. Second, a higher redundancy ratio for the Tetrys sender leads to less delay recovery time for lost packets since the inter-arrival time between two consecutive repair packets is shortened.

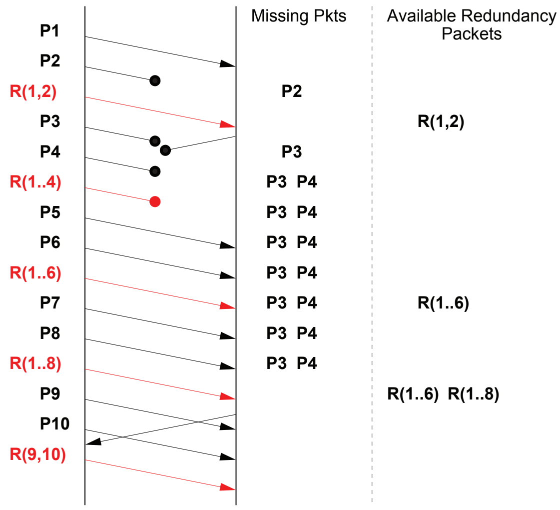


Figure 1: A simple data exchange with Tetrys ($k = 2$) [9].

3. Redundancy adaptation algorithm for real-time video transmission

This section first introduces our previous work which investigated the model on packet recovery delay. Then, we present a redundancy adaptation algorithm for real-time video transmission which adapts to network dynamics based on insights from this previous work.

3.1. Previous work

In [9], Tournoux et al. proposed a heuristic model $\theta(t)_{(d,p,b,T,R)}$ (see the notations in Table 1) for multimedia applications that requires an arrival of a certain amount of packets within a tolerable delay constraint D_{max}. This model gives the cumulative distribution function of recovery delay of lost packets. The model assumes a Constant Bit Rate (CBR) with the same packet size that produces a data packet every T seconds based on a network state. The authors found that $\theta(t)_{(d,p,b,T,R)}$ fits well to the Weibull distribution which is defined by the scale λ and the shape κ parameters as follows:

$$P[X < x] = 1 - e^{-(x/\lambda)^{\kappa}} \tag{1}$$

The Weibull function in Eq. (1) is applied to Tetrys with the parameters $\lambda(\Delta_R)$ and $\kappa(\Delta_R)$. $\lambda(\Delta_R)$ is inversely proportional to Δ_R and is expressed

Table 1: Notations

k	The number of sent source packets between two consecutive repair packets
n	The total number of source packets plus a repair packet $n = k+1$
R	Redundancy ratio $R = \frac{1}{n}$
p	Packet loss rate
b	Average length of consecutive lost packets (mean burst size)
Δ_R	The difference between redundancy ratio and packet loss rate $\Delta_R = R - p = \frac{1}{n} - p$
d	The propagation delay
D_{max}	The maximum tolerable delay required by the application
T	The mean interval time between two consecutive source packets
I	The mean interval time between two consecutive repair packets
y	The number of lost packets needing to be recovered in the receiver buffer
z	The number of repair packets received at the receiver
Z	The number of additional repair packets needed to recover all losses $Z = y - z$
P_i	The first lost packet which has not been recovered yet since last packet recovery event
t_i	The remaining time to recover the first lost packet (as well as all lost packets) before the deadline D_{max}

as $\lambda(\Delta_R) = \frac{a_\lambda}{\Delta_R{}^{b_\lambda}}$ while $\kappa(\Delta_R)$ evolves linearly as a function of Δ_R and is expressed as $\kappa(\Delta_R) = a_\kappa * \Delta_R + b_\kappa$. The coefficients a_i, b_i ($i \in \{\lambda, \kappa\}$) are related to the loss pattern (p and b) and n. The values of these coefficients can be found in [9]. The Weibull function applied to Tetrys returns a recovery probability of lost packets before a deadline D_{max} given <u>*i)*</u> a network state specified by a delay d, a packet loss rate p and a burstiness of losses b <u>*ii)*</u> a redundancy ratio $R = 1/n$. This heuristic model has some drawbacks. First, it requires an accurate channel estimation, which is not an obvious task. Furthermore, this model does not adapt well to network changes where both the loss rate, the burstiness of losses and the propagation delay vary over time. However, this model does give us some insights designing a redundancy adaptation algorithm presented in Section 3.2. In this article, we introduce briefly the model for later explanations in the algorithm, the original paper

[9] gives further details.

3.2. Redundancy adaptation algorithm

The redundancy adaptation algorithm based on Tetrys aims to minimize the impact of packet losses in the context of real-time video transmission. Indeed, the algorithm seeks to answer the two following questions: 1) Which criteria are necessary to increase redundancy? 2) Which criteria are used to decrease redundancy? Before answering these questions, we give an overall view of the Adaptive Tetrys framework shown in Fig. 2 for real-time video transmission. The video encoder encodes the live source video based on the quality/redundancy controller. Then, the Tetrys encoder takes the encoded video and creates linear combinations for the repair packets according to the current redundancy ratio. At the receiver side, the Tetrys receiver tries to decode the lost packets and pass the recovered lost packets to the video decoder as soon as possible. The monitoring agent observes the loss pattern and delay induced by the network. The redundancy adaptation module gathers the information from the monitoring agent and sends increasing redundancy feedback, decreasing redundancy feedback or does nothing according to the algorithm presented below. The sender receives the feedback information and changes the redundancy ratio and video quality accordingly.

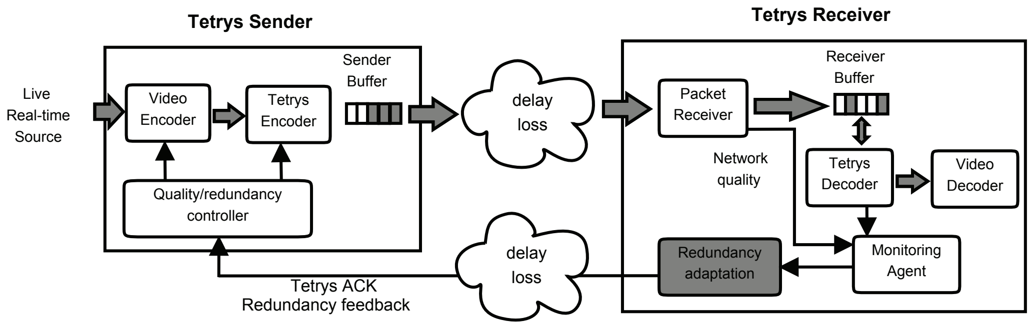


Figure 2: Adaptive Tetrys framework for real-time video transmission.

3.2.1. Which criteria are necessary to increase redundancy?

In Section 2, we noted that the first lost packet (as well as all lost packets) can be recovered when $Z = 0$. This means that the number of received repair packets is equal to the number of lost packets. When the Tetrys receiver observes some lost packets that have not been recovered yet (i.e., $Z > 0$), it estimates the arrival time of the first lost packet P_i in the absence of losses based on T and the arrival time of the successfully received previous packet P_{i-1}. The Tetrys receiver then deduces the remaining time t_i to recover packet P_i as well as all lost packets before the deadline D_{max} from the estimated arrival time of the packet P_i and D_{max}. In an ideal case where there are no further losses for both data and repair packets, the Tetrys receiver needs $Z * I$ (in time) to recover all losses. The condition $Z * I < t_i$ implies that all losses can be recovered before the application constraint D_{max} while $Z * I > t_i$ implies that some lost packets cannot be recovered before the application deadline. However, the algorithm actually needs $Y \geq Z$ to recover all losses, since losses may still occur up until the time when the receiver receives enough Tetrys repair packets. Y depends on the loss distribution (e.g., Bernoulli or Gilbert-Elliott [13]). In [9], Tournoux et al. theoretically calculate the decoding delay knowing Z for the case of Bernoulli where the losses are uniformly distributed. However, the implementation is far from being trivial. Furthermore, there are no theoretical estimations of the decoding delay for other loss patterns (e.g., Gilbert-Elliott). Thus, we propose building an algorithm that increases the redundancy ratio if either of the two following conditions is not satisfied:

1. $Z * I * f < t_i$
2. $P[X < t_i] \geq th_{min}$

where $f > 1$ is a coefficient that indicates the proactive level of the algorithm. A larger f value means that the algorithm is more proactive to react to packet losses by adapting quickly to the redundancy before exceeding the application delay constraint and vice versa. The first condition implies a reactive behavior that the receiver actually observes at a given time, while the second condition indicates an estimation behavior that might occur in the future. In fact, given t_i, p and b observed at the receiver, the algorithm increases the redundancy if the probability from the Weibull function in Eq. (1) to recover the lost packets before t_i is lower than a certain threshold th_{min} (e.g., 0.9) which is required for the applications. When either of the two conditions is not satisfied, the Tetrys receiver sends a feedback message to the Tetrys sender to require a redundancy increment.

3.2.2. Which criteria are used to decrease redundancy?

The Tetrys receiver sends a feedback message that requires a redundancy decrement if both of the following conditions are satisfied:

1. $Z = 0$
2. $P[X < D_{max}] \geq th_{max}$

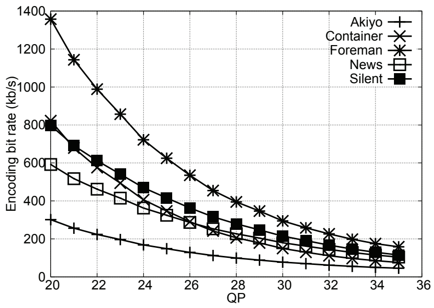
The first condition means that at a given time, there are no unrecovered packets. The second condition indicates that with the current redundancy ratio and the observed network state, the probability from the Weibull function of recovering packet losses before the application deadline D_{max} is greater than a certain threshold th_{max} (e.g., 0.99). Thus, these two conditions allow the safe reduction of redundancy. It is obvious that th_{max} must be greater than th_{min}. The impact of the difference between th_{min} and th_{max} is studied in Section 5.1.

3.3. Feedback information in Tetrys acknowledgment

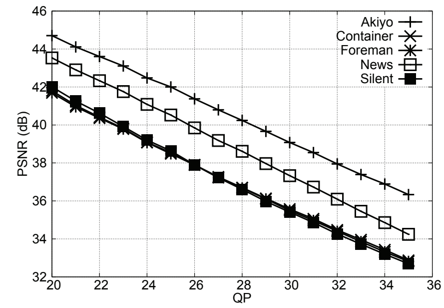
According to the algorithm, the Tetrys receiver sends a feedback message each time it requires a redundancy increment or decrement. These feedback messages might be lost during transmission. The loss of a feedback message that requires a redundancy decrement does not have much impact on the residual loss rate since the Tetrys sender uses a much higher redundancy ratio than the current loss rate. However, the loss of feedback message that requires a redundancy increment has a stronger impact on performance since the Tetrys receiver experiences packet losses that might not be recovered within the application time constraint. Furthermore, the losses may still persist or even become worse. This leads to more lost packets that cannot be recovered before the application deadline. In a case where all increasing redundancy feedback messages are lost, A-Tetrys turns out to be standard Tetrys where the redundancy ratio is not changed regardless of network conditions. Thus, we propose a simple mechanism which is more robust to feedback losses. Indeed, in the event the Tetrys receiver decides to send a feedback message (redundancy increment or decrement), it sends a Tetrys acknowledgment packet in which the feedback information is included. This feedback information is also included in the periodic Tetrys acknowledgment packets afterwards until the Tetrys sender updates its redundancy ratio. The Tetrys sender only updates its redundancy once when it first sees the update requirement. In this way, Tetrys does not need to handle a new packet type.

4. Redundancy list for H.264/AVC real-time transmission

The redundancy adaptation algorithm in Section 3.2 does not specify the amount of redundancy adjustment. In general, the n parameter of Tetrys only takes integer values $n \in \{2, 3, 4, 5, 6, ...\}$ which is equivalent to the list of redundancy ratios $R \in \{0.5, 0.33, 0.25, 0.2, 0.17, ...\}$. However, this general redundancy list may not fit well to video transmission where the video characteristics are taken into account.



(a) Encoding bit rate for different video sequences



(b) PSNR for different video sequences

Figure 3: Different CIF video sequences encoded by x264 with Baseline profile.

In video coding, the quantization parameter (QP) controls the trade-off between data rate and image quality [14]. Indeed, the QP is inversely proportional to the image quality. From [15, 16], the relationship between rate and QP can be modeled by the following equation

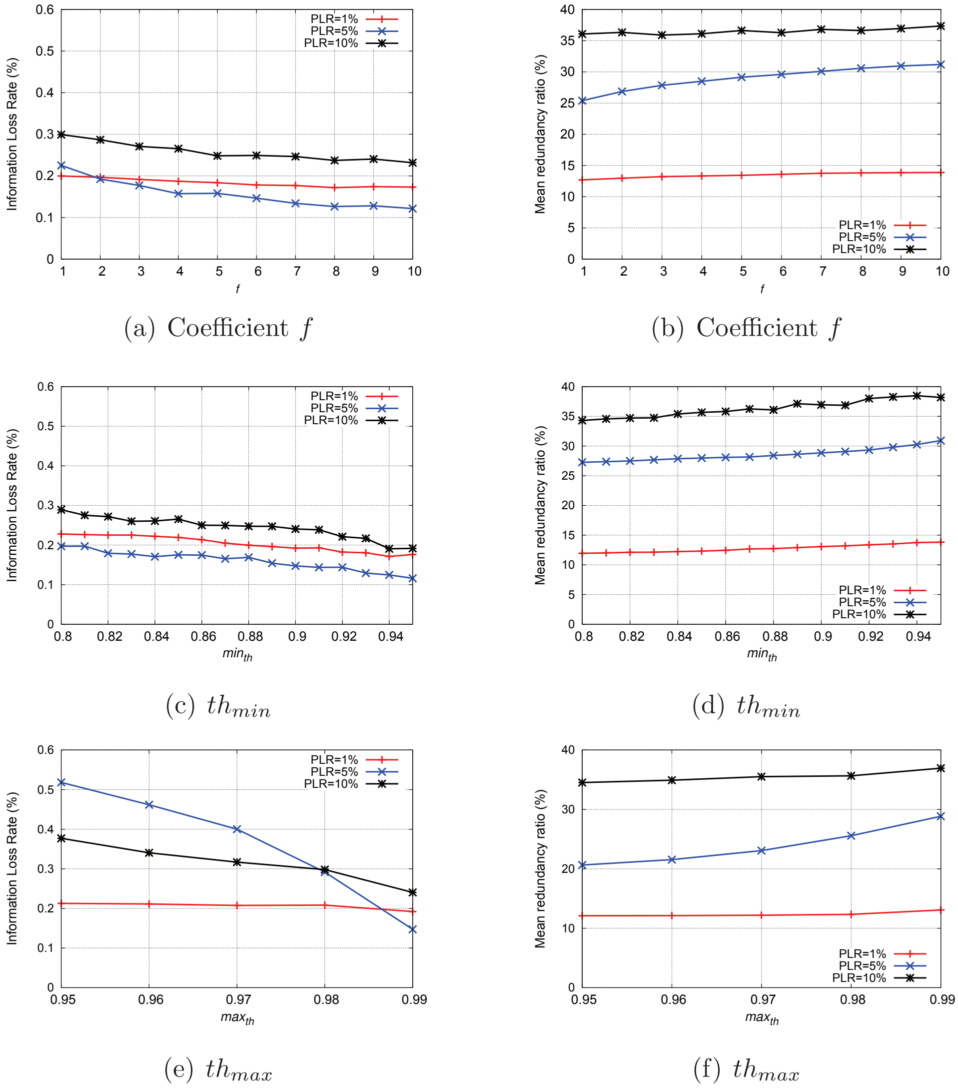
$$R_{video} = \alpha * e^{\beta * QP} \tag{2}$$

where coefficients α and β ($\beta < 0$) specify the video characteristics. The model in Eq. (2) fits well to the experiments from x.264 encoder [17] (see Fig. 3). Quantitatively, each time the value of QP is increased by one, the encoding bit rate gain is in the range of 10% to 20% while the video quality degradation is in the range of 0.5 to 1 dB. This percentage gain in bit rate can be used by erasure codes to protect the video from losses. It should be noted that the impact of a slightly degraded video ranging from 0.5 to 1 dB is negligible to the human eye. Similar results with different video encoders (e.g., JM [18], x264), video profiles (e.g., Baseline, High), video formats (e.g., QCIF, 4CIF, 720p), Group of Pictures (GOP) sizes and QP patterns can be found in [19]. Thus, we propose a redundancy list for the case of H.264/AVC video transmission $R \in \{0.1, 0.2, 0.33, 0.5\}$ which is equivalent to the list for $n \in \{10, 5, 3, 2\}$. The chosen list of redundancy ratios ensures that the lower quality video plus redundancy used by Tetrys does not send extra bit rate compared to the normal quality video without protection. This prevents the possibility of congestion caused by the extra bit rate injected on to the networks. Let us give an example by assuming that the Tetrys sender is transmitting a video with $QP = 29$ and a Tetrys redundancy ratio of 20%. On the one hand, if the Tetrys sender receives an increasing redundancy feedback, the Tetrys sender increases its redundancy ratio to 33.3% while decreasing the video quality to $QP = 30$. On the other hand, if the Tetrys sender receives decreasing redundancy feedback, the Tetrys sender reduces its redundancy ratio to 10% while increasing the video quality to $QP = 28$. In a case where the Tetrys sender receives a decreasing redundancy feedback while its redundancy is 10%, the Tetrys sender maintains its redundancy ratio since 10% is the lowest value in its redundancy list and it is necessary to protect the video from packet losses.

5. Evaluating the algorithm parameters with CBR traffic

We evaluate A-Tetrys using the network simulator ns-2 [20]. We send a Constant Bit Rate traffic at $1900\,kb/s$ with a constant packet size of 500 bytes. The one-way propagation delay is set to $50\,ms$ which results in a $100\,ms$ Round Trip Time (RTT) and the one way end-to-end (E2E) delay constraint D_{max} is set to $150\,ms$, based on ITU-T/G.144 [21]. This constraint is recommended for highly interactive applications. The packets recovered after this deadline are considered as lost by the application. The Tetrys acknowledgment frequency is set to $10\,ms$. The acknowledgment packet which has a small size compared to information or repair packet is periodically sent on the reverse path. Indeed, the size of Tetrys acknowledgment packet is 36 bytes (including 28 bytes of UDP/IP header). Theoretically, the data rate of the feedback channel is less than $30\,kb/s$ for a feedback frequency of $10\,ms$, which is quite low compared to the data rate on the forward channel. Thus, the bandwidth occupation is not critical and can be neglected. We evaluate the performance using the Information Loss Rate (ILR) which indicates the residual loss rate after decoding within the application deadline at the end of each simulation. Tetrys shows best performance against uniform losses

[9, 22], thus, we only evaluate the performance with the Gilbert-Elliott erasure channel which is specified by an average Packet Loss Rate (PLR) and an average length of consecutive lost packets (or mean burst size) [13]. To provide a fair comparison, the sender sends the same number of data packets (50000) while the number of repair packets depends on the redundancy ratio used in each simulation.



(a) Coefficient f

(b) Coefficient f

(c) th_{min}

(d) th_{min}

(e) th_{max}

(f) th_{max}

Figure 4: Impact of the algorithm parameters at mean burst size $b = 3$.

5.1. Impact of algorithm parameters

We first evaluate the impact of coefficient f by disabling the second condition ($P[X < t_i] \geq th_{min}$) in the increasing redundancy criteria. The th_{max} is set to 0.99 in the decreasing redundancy criteria. Fig. 4(a) shows a slight decreasing trend in ILR for different PLRs with mean burst size $b = 3$ when the coefficient increases. The decrease in ILR leads to an increase in the average redundancy ratio which is shown in Fig. 4(b). The greater coefficient f implies a more proactive approach against packet losses and vice versa. It is notable that the ILR of PLR = 5% is smaller than PLR = 1%. This can be explained by the amount of redundancy used by A-Tetrys in both simulations. In fact, at $f = 3$, A-Tetrys uses on average ≈13% during the simulation at PLR = 1% while it uses on average ≈28% at PLR=5%. Furthermore, since the chosen redundancy list $R \in \{0.1, 0.2, 0.33, 0.5\}$ (i.e., $n \in \{10, 5, 3, 2\}$) is specified for video transmission, this implies that an important amount of redundancy (>= 10%) is added or removed for every change in redundancy ratio. The result is different if the general redundancy list is chosen $R \in \{0.5, 0.33, 0.25, 0.2, 0.17, ...\}$ (i.e., $n \in \{2, 3, 4, 5, 6, ...\}$) where the amount of redundancy change is finer.

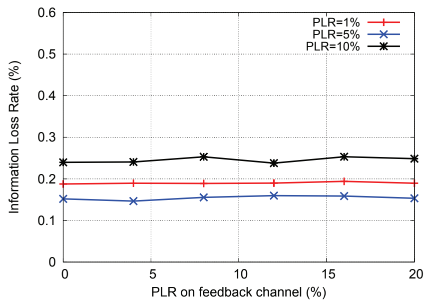
Then, we evaluate the impact of th_{min} by disabling the first condition ($Z * I * f < t_i$) in the increasing redundancy criteria. The th_{max} is still set to 0.99 in the decreasing redundancy criteria. Fig. 4(c) shows that the greater value of th_{min} results in a lower ILR. The remark for PLR=1% and PLR=5% at $b = 3$ is similar to the case of coefficient f. Furthermore, at PLR=5% and $b = 3$, the redundancy ratio of Tetrys is greater than or equal to 20% most of the time since the second condition in the increasing redundancy criteria is not satisfied if the redundancy ratio is 10% compared to PLR=5% and $b = 3$ (Fig. 4(d)).

Finally, we evaluate the impact of th_{max} by setting $f = 2$ and the $th_{min} = 0.9$. Fig. 4(e) shows that the algorithm experiences a higher ILR if the th_{max} is low. Indeed, the lower value of th_{max} implies a closer gap between $th_{min} = 0.9$ and th_{max} where the algorithm changes frequently its redundancy ratio. In fact, a redundancy ratio of 10% is not high enough to cover a PLR of 10%. Thus, the algorithm switches the redundancy ratio between 20% and 33.3% most of the time, while the switch between the redundancy ratios between 10% and 20% is frequent at PLR=5%. For instance, at $th_{max} = 0.95$, the redundancy ratio switches frequently since th_{max} is close to $th_{min} = 0.9$. This explains why the ILR at PLR=5% is lower than the one at PLR=10% when

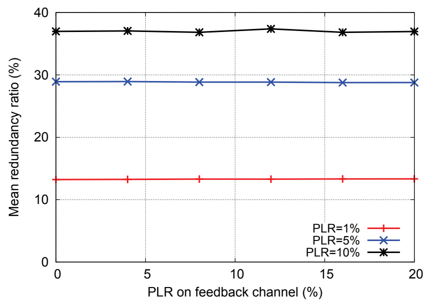
th_{max} is low. Thus, we recommend using a reasonable value of th_{min} that is required for applications and a high value of th_{max} (i.e., greater than 0.98).

5.2. Impact of losses on feedback channel

To evaluate the impact of losses on the feedback channel, we conducted the same simulations as in Section 5.1 with these settings: $f = 2$, $th_{min} = 0.9$ and $th_{max} = 0.99$. We use a Bernoulli erasure channel for the feedback link. The loss pattern on the forwarding path is the same as the previous simulations. From Fig. 5(a), we see that the ILR curve is rather flat against the increasing loss rate on the feedback channel. These simulations show that the algorithm is robust to the loss rate on the feedback channel by including the feedback information in the Tetrys acknowledgment packets as presented in Section 3.3.



(a) Information loss rate



(b) Mean redundancy ratio

Figure 5: Impact of losses on the feedback channel. The forward channel has a mean burst size $b = 3$. The loss rate and mean burst size on the forwarding channel is independent of the feedback channel.

6. Evaluation with video traffic

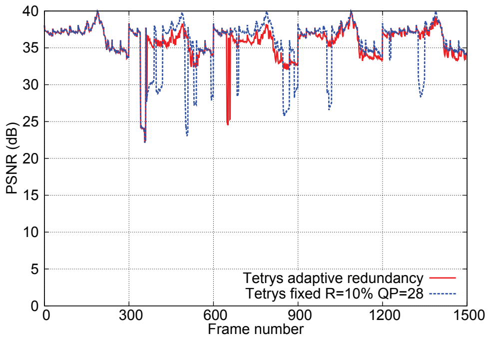
The one-way propagation delay, the one-way E2E delay constraint and the Tetrys acknowledgment frequency are set as in Section 5. The 'Foreman' CIF video sequence of 300 frames is repeated 5 times to provide a video of 1500 frames at a rate of 30 frames per second. This results in 50 seconds of real-time video transmission. Thus, each 10 seconds of simulation represents a single 'Foreman' sequence. We encode the video using basic coding where there is no error resilience mechanism (e.g., Flexible Macroblock Ordering, etc.) [3, 4]. The GOP size is set to 30 images and the packet size varies and depends on the encoded video. The repair packet takes the maximum size among all information packets it is constructed from. The video is encoded using the Baseline profile which is suitable for real-time video transmission. The loss concealment mechanism is frame copy. We set the coefficient $f = 4$, $th_{min} = 0.9$ and $th_{max} = 0.99$. The video is evaluated with both PSNR and Continuity Index (CI). The CI is defined as the ratio of the number of continuous frames decoded without errors (not including the distorted frames caused by error propagation) to the total number of frames. We evaluate the videos with three schemes: A-Tetrys, standard Tetrys and without Tetrys protection. The video without Tetrys protection is encoded with $QP = 27$ while the video with a fixed Tetrys redundancy ratio of 10% is encoded with $QP = 28$. The QP in the video protected by A-Tetrys varies according to the redundancy ratio in such a way that the bandwidth occupation does not exceed the video without Tetrys protection. We evaluate all three scenarios. In the first scenario, the loss rate is fixed while the mean burst size varies. Both loss rate and mean burst size vary in the second scenario. Finally, both loss rate and mean burst size are fixed while the RTT varies in the third scenario.

6.1. Evaluation with fixed loss rate and variable mean burst size

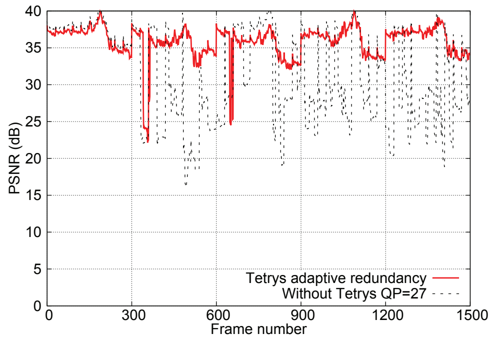
Table 2: Loss pattern during 50 s of simulation in Section 6.1

Time (s)	Loss pattern	Frame range
0 - 10	no losses	0 - 300
10 - 30	Gilbert-Elliott PLR=2%, $b = 2$	301 - 900
30 - 50	Bernouilli PLR=2%	901 - 1500

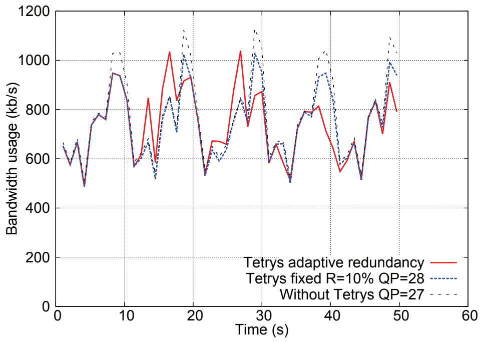
The loss pattern over 50 seconds of simulation is shown in Table 2. Fig. 6(a) shows the results between A-Tetrys and standard Tetrys. In the frame range from 0 to 300, the PSNR of A-Tetrys is the same as that of standard Tetrys since there are no losses. The A-Tetrys maintains its minimum redundancy ratio of 10%. In the frame range from 301 to 900 where the Gilbert-Elliott loss pattern with PLR=2% and $b = 2$ occurs, standard Tetrys observes a more significant drop in quality than A-Tetrys. In some frames, A-Tetrys has a slightly lower PSNR in the absence of video quality



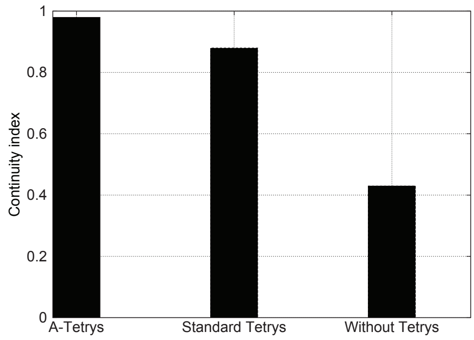
(a) A-Tetrys vs standard Tetrys



(b) A-Tetrys vs without Tetrys protection



(c) Instantaneous bandwidth usage



(d) Continuity index

Figure 6: Comparison between 3 schemes with fixed PLR and variable b.

degradation. This is because A-Tetrys lowers the video quality by increasing the QP for more redundancy to adapt to network conditions. However, visually, this slightly lower quality cannot be clearly distinguished by the human eye. However, the end users experience much stronger impact in each event where the PSNR significantly drops due to residual packet losses. In the frame range from 901 to 1500 where the Bernoulli loss pattern with PLR=2% occurs, standard Tetrys performs well. It gets only one quality degradation event while A-Tetrys does not experience any residual losses. Fig. 6(b) shows the poor performance of the video without protection by Tetrys regardless of the loss patterns. Fig. 6(d) shows that A-Tetrys has a very high CI of 0.98 while standard Tetrys and the video without protection have a CI of 0.88 and 0.43, respectively. Fig. 6(c) shows the bandwidth usage observed at the outgoing interface of the sender, it can be seen that all three schemes use similar bandwidth on average. Table 3 shows that A-Tetrys objectively gains on average only 0.2 dB compared to standard Tetrys; but subjective evaluation by watching the resulting videos [19] and Fig. 6(a) shows a much better performance by A-Tetrys. Additionally, A-Tetrys and standard Tetrys both achieve the same PSNR in first 10 seconds of simulation since there are no losses. This explains why the objective evaluation does not always adequately reflect the video quality experienced by the end users. It should be noted that the standard deviation of A-Tetrys which indicates a fluctuation in video quality is much lower than the one of standard Tetrys. Table 3 also shows that the video with A-Tetrys uses less bandwidth on average than the video without Tetrys protection. This confirms our conservative choice of redundancy ratio list in Section 4 where the video with A-Tetrys does not use more bandwidth than the video without protection.

Table 3: Mean and standard deviation of PSNR and bandwidth usage with different schemes in Section 6.1

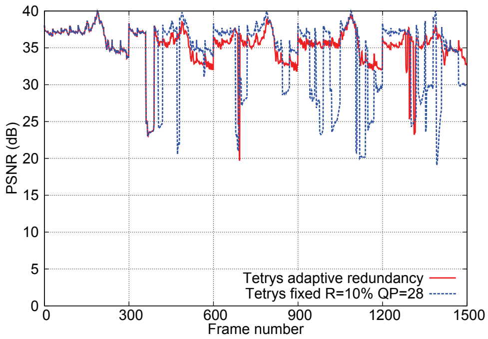
	PSNR (dB)	BW usage (kb/s)
A-Tetrys	35.9 ± 2.3	737.8 ± 140.3
Standard Tetrys	35.7 ± 3.3	740.3 ± 148.7
Without Tetrys	31.1 ± 6.4	774.1 ± 174.8

6.2. Evaluation with both variable loss rate and mean burst size

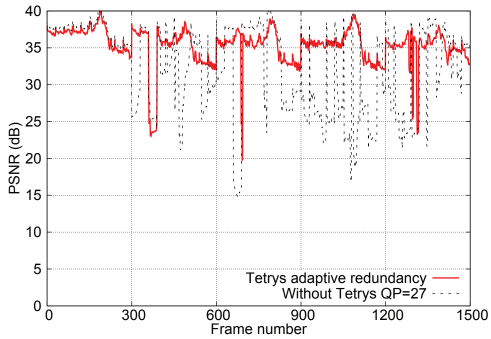
Table 4: Loss pattern during 50 s of simulation in Section 6.2

Time (s)	Loss pattern	Frame range
0 - 10	no losses	0 - 300
10 - 20	Gilbert-Elliott PLR=2%, $b = 2$	301 - 600
20 - 30	Gilbert-Elliott PLR=2%, $b = 3$	601 - 900
30 - 40	Gilbert-Elliott PLR=3%, $b = 2$	901 - 1200
40 - 50	Gilbert-Elliott PLR=3%, $b = 3$	1201 - 1500

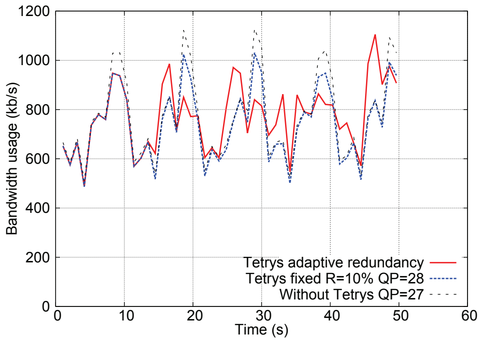
The loss pattern in this simulation is shown in Table 4. Fig. 7(a) shows that standard Tetrys exhibits the performance problem from variations of



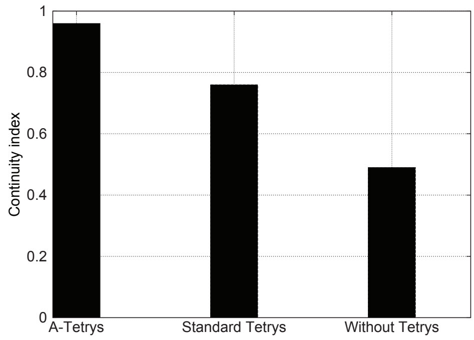
(a) A-Tetrys vs standard Tetrys



(b) A-Tetrys vs without Tetrys protection



(c) Instantaneous bandwidth usage of 3 schemes



(d) Continuity index

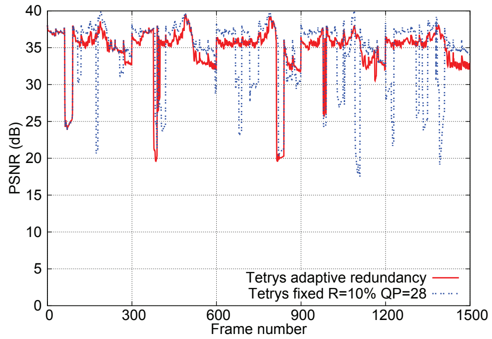
Figure 7: Comparison between 3 schemes with varied both PLR and b.

both PLR and mean burst size. In fact, when the PLR is increased from 2% to 3% from frame 901, standard Tetrys experiences more residual losses than previous frames which leads to more video quality degradation events. Fig. 7(b) confirms that video without protection from erasure codes exhibits poor performance from both PLR and mean burst size. Fig. 7(d) shows that A-Tetrys, standard Tetrys and the video without protection achieve a CI of 0.96, 0.73 and 0.49, respectively. The instantaneous bandwidth usage of A-Tetrys in Fig. 7(c) is slightly different from Fig. 6(c) since it uses both different redundancy ratio and video quality to adapt to the network state. Table 5 shows that A-Tetrys objectively gains on average 1.2 dB compared to standard Tetrys. Furthermore, from the subjective evaluation perspective, A-Tetrys gives a much better performance [19]. In this simulation, the video with A-Tetrys uses the same average bandwidth as the video without Tetrys protection.

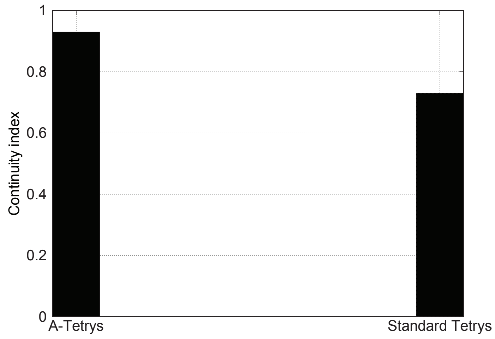
Table 5: Mean and standard deviation of PSNR and bandwidth usage with different schemes in Section 6.2

	PSNR (dB)	BW usage (kb/s)
A-Tetrys	35.3 ± 2.6	$773.8.3 \pm 138.1$
Standard Tetrys	34.1 ± 5.0	740.3 ± 148.7
Without Tetrys	31.9 ± 6.2	774.1 ± 174.8

6.3. Evaluation with varied Round Trip Time



(a) PSNR



(b) Continuity index

Figure 8: A-Tetrys vs standard Tetrys with varied RTT.

The loss pattern is fixed to PLR=2% and $b = 2$ during the simulation. The one-way propagation delay is set to $50\,ms$ at the beginning of the simulation and increases to $70\,ms$ after 20 seconds. Fig. 8(a) shows that both A-Tetrys and standard Tetrys perform well at the one-way delay of $50\,ms$. However, when the delay is increased to $70\,ms$ where the remaining time to recover the lost packets is shortened, standard Tetrys observes a greater drop in quality. On the other hand, the performance of A-Tetrys remains constant since the algorithm takes into account this change from the signal t_i and reacts accordingly. Indeed, A-Tetrys has a CI of 0.93 while standard Tetrys obtains a CI of 0.73 (Fig. 8(b)). These three simulations show that A-Tetrys consistently achieves a CI greater than 0.9.

7. Comparison with FEC adaptation scheme

While Tetrys adapts to network dynamics by changing only one parameter, the redundancy ratio, redundancy adaptation with FEC is more complicated. In this article, FEC denotes the Reed-Solomon FEC code in [6]. First, FEC(k,n) which indicates k source packets and $n - k$ repair packets requires changing both the group size n and the redundancy ratio $(n-k)/n$. It is not evident to provide the largest group size possible before transmitting the data since FEC is more robust to burstiness losses at a larger group size. However, a large group size may lead to inefficiency since FEC repair packets may arrive after the application delay constraint due to its group size or a longer delay caused by the network. Second, the criteria for adapting the FEC redundancy ratio and group size are not obvious. Tetrys has the signals from the first lost packet which has not been recovered yet and the probability of recovering the losses before the delay constraint. On the contrary, FEC must wait for the arrival of the last packet in a FEC group if it is unable to recover the lost packets with the current received packets.

In order to provide some insights into how A-Tetrys performs compared to FEC, we propose a simple redundancy algorithm for FEC with the assumption that the best FEC group size n is known. The redundancy ratio list is the same as with Tetrys ([10, 20, 33.3, 50]%). The algorithm decides to increase the redundancy if its current redundancy ratio is less than the observed loss rate plus a threshold min_{FEC} mathematically presented by $R_{FEC} < p + min_{FEC}$. Similarly, the algorithm decreases the redundancy if $R_{FEC} > p + max_{FEC}$. In this case, min_{FEC} must be lower than max_{FEC}.

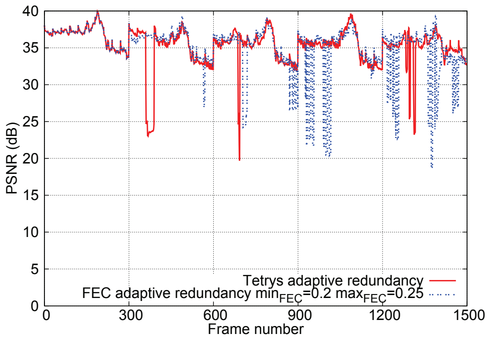
We conducted several simulations to determine the largest FEC group size (i.e., the best FEC group size) that would not be inefficient. By varying the FEC group size in each simulation, we found that the largest FEC group size is 10 packets. Thus, we set n at or close to 10 for the simulation and let k vary according to the redundancy ratio. For instance, if the redundancy ratios are 10% and 33.3%, we use FEC(9,10) and FEC(6,9), respectively. It can be noted that the FEC group size can be larger with higher quality or video resolution (e.g., 4CIF or 720p) where there are more packets per image encoded than the CIF 'Foreman' video. We used the loss pattern as in Table 4. We varied min_{FEC} from 0.06 to 0.2 with a step size of 0.02 and max_{FEC} from 0.1 to 0.3 with a step size of 0.05 while satisfying the constraint $min_{FEC} < max_{FEC}$. We chose the combination where $max_{FEC} = 0.25$ and $min_{FEC} = 0.2$ that provides the best performance to compare to A-Tetrys.

The performance evaluation is based on the number of decoded frames which have a PSNR greater than 30 dB. Fig. 9(a) shows that at PLR=2% with both $b = 2$ and $b = 3$ where the video frame ranges between 301 and 900, FEC with adaptive redundancy achieves similar performance to A-Tetrys. However, when the PLR is increased to 3% from frame 901, we see that FEC exhibits higher video quality degradation than A-Tetrys. Furthermore, from frame 1201 where the mean burst size is equal to 3, FEC experiences severe quality degradation due to residual losses. Since the FEC group size is small, FEC exhibits more problems at higher burst sizes. From the simulation, FEC with adaptive redundancy uses an average redundancy ratio of 26.2%. To compare with traditional FEC, we also conduct a simulation with FEC(7,10) without adaptive redundancy which resulted in a redundancy ratio of 33.3%. Even though the redundancy ratio of 33.3% is favorable to FEC, Fig. 9(b) shows that A-Tetrys still provides better performances than FEC. Table 6 shows that Tetrys achieves a better PSNR and uses less bandwidth than FEC with and without adaptive redundancy. Additionally, Fig. 9(c) shows that both FEC with adaptive redundancy ratio and FEC(7,10) have a CI of 0.89 while A-Tetrys obtains an value of 0.96.

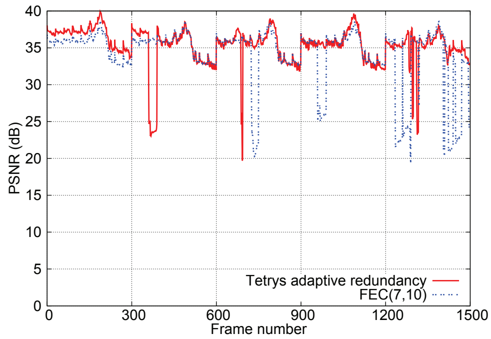
This section aims to test A-Tetrys against a FEC redundancy adaptation algorithm. However, as we are not able to ensure that the FEC redundancy adaptation algorithm provides the best possible configuration, we propose to complete the comparison with the assumption that the best FEC block size is known. The results show that A-Tetrys outperforms the proposed adaptive FEC even though the comparison approach is in favor of FEC (probing the best block size before starting video transmission, choosing the best combination of adaptive FEC parameters max_{FEC} and min_{FEC}). In [9, 22], with fixed redundancy ratio, we have shown that Tetrys outperforms FEC in both single path and multipath transmissions.

Table 6: Mean and standard deviation of PSNR and bandwidth usage with different schemes in Section 7

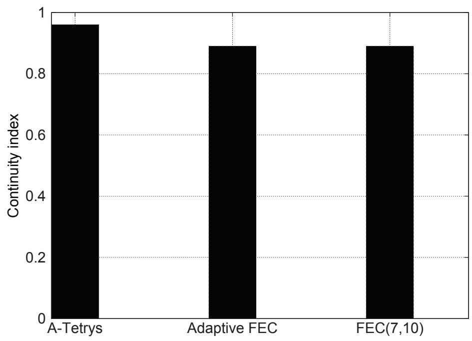
	PSNR (dB)	BW usage (kb/s)
A-Tetrys	35.3 ± 2.6	773.8 ± 138.1
FEC with adaptive redundancy	35.0 ± 3.5	889.3 ± 155.1
FEC without adaptive redundancy	34.1 ± 4.1	896.5 ± 136.5



(a) A-Tetrys vs FEC with adaptive redundancy algorithm



(b) A-Tetrys vs FEC with a fixed redundancy ratio of 33.3%



(c) Continuity index

Figure 9: A-Tetrys vs FEC.

8. Related work

Our approach differs from the existing work in the following aspects. First, we use an on-the-fly and systematic erasure network coding scheme that shows better performance than FEC codes in terms of packet recovery rate in both single-path and multi-path transmissions [9, 22]. Although Tetrys uses an elastic encoding window buffer to construct the repair packets, Tetrys sends a repair packet for every k information packets. Thus, it can be considered that Tetrys belongs to the class of rateful codes. Indeed, Tetrys has a code rate of $k/(k+1)$. Additionally, the stringent time constraint in real-time video transmission limits the total number of information and repair packets (i.e., n). One of the important advantages of rateless codes such as Fountain or LT codes [23, 24] is that these codes can generate a potential infinite number of encoded symbols from k information symbols[1]. The rateless codes are known to be less efficient than Maximum Distance Separable (MDS) codes for a fixed or limited block length. Therefore, it is not beneficial to use rateless codes in this context when n is limited. Therefore, we do not compare with the work using rateless codes.

Second, A-Tetrys focuses real-time video transmission with a stringent delay constraint required by applications such as video conferencing while the existing proposals target the context where the receiver has a large playout buffer [10, 25]. In case of broadcast channels or distributed streaming, randomized linear codes show their benefits [26, 27, 28]. On the other hand, our work focuses on point-to-point and/or point-to-multipoint where systematic erasure codes show good performance, especially for video transmission with stringent delay constraint. MPEG-DASH [29, 30] is standardized in 2012 for dynamic adaptive streaming over HTTP which usually employs TCP/IP as underlying protocol. TCP is a reliable transport protocol which asks for retransmission of lost packets. The delay to recover the lost packets requires at least one additional RTT which is not applicable for the real-time video transmission with a hard deadline.

Third, the work on joint source channel coding usually tries to minimize the distortion induced from source error caused by video compression and channel error caused by packet losses [31, 32]. On the other hand, our work does not aim at minimizing the distortion but the residual packet loss rate since the unrecovered lost packets have much visual impact. The authors in [33] propose a new joint source-channel approach for adaptive FEC. Their scheme is similar to our simple FEC adaptation scheme which is proposed for the comparison with A-Tetrys. In fact, they set the block size n and try to find the optimal code rate according to channel loss rate.

Lastly, our algorithm does not add extra bit rate by exploiting the relationship between the redundancy ratio and the variation of the Quantization Parameter [34]. In [35], the authors propose a FEC redundancy adaptation algorithm inside the Encoded Multipath Streaming (EMS) scheme. This algorithm increases the redundancy ratio if the residual loss rate after decoding is greater than a certain threshold and vice versa. Our approach is to minimize the residual loss rate to increase the video quality experienced by end

[1]In this paper, the encoded symbols are considered as the information and repair packets which is specified by n

users. Furthermore, the redundancy adjustment in [35] is not video-aware while our algorithm adjusts the redundancy ratio based on the changes in the Quantization Parameter.

9. Conclusions

In this paper, we introduced a redundancy adaptation algorithm based on an on-the-fly erasure network coding scheme for real-time video transmission called Tetrys. By exploiting the relationship between the changes in the Quantization Parameter, the loss or gain in encoding bit rate and the Tetrys redundancy ratio, a video with A-Tetrys achieves better video quality in terms of PSNR than both the video with standard Tetrys and the video without Tetrys protection. We chose the redundancy ratio list so that the video with A-Tetrys does not send more bit rate than the video without projection to prevent congestion. We have shown that A-Tetrys performs well with the variations of both loss pattern and delay induced by networks. Finally, we also showed that A-Tetrys outperforms FEC with and without redundancy adaptation.

10. Acknowledgments

This work was supported by the French ANR grant ANR-VERS-019-02 (ARSSO project). The authors would like to thank Dr. Pierre-Ugo Tournoux for useful discussions.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, Overview of the H.264/AVC Video Coding Standard, IEEE Transactions on Circuits Systems for Video Technology 13 (7) (2003) 560–576.

[2] G. Sullivan, J. Ohm, W.-J. Han, T. Wiegand, Overview of the High Efficiency Video Coding (HEVC) Standard, IEEE Transactions on Circuits and Systems for Video Technology 22 (12) (2012) 1649–1668.

[3] S. Wenger, H.264/AVC over IP, IEEE Transactions on Circuits and Systems for Video Technology 13 (2003) 645–656.

[4] S. Kumar, L. Xu, M. K. Mandal, S. Panchanathan, Error Resiliency Schemes in H.264/AVC Standard, J. Visual Communication and Image Representation 17 (2) (2006) 425–450.

[5] T. T. Tran, J. Lacan, H. Meric, Error tolerance schemes for H.264/AVC: An evaluation, in: Proceedings of IEEE Consumer Communications and Networking Conference (CCNC), 2012, pp. 571 –575.

[6] J. Lacan, V. Roca, J. Peltotalo, S. Peltotalo, Reed-Solomon Forward Error Correction (FEC) Schemes, RFC 5510 (Apr 2009).

[7] J. Kumar Sundararajan, D. Shah, M. Medard, ARQ for network coding, in: IEEE International Symposium on Information Theory, 2008, pp. 1651 –1655.

[8] J. Lacan, E. Lochin, Rethinking reliability for long-delay networks, in: Proceedings of International Workshop on Satellite and Space Communications IWSSC, 2008, pp. 90–94.

[9] P.-U. Tournoux, E. Lochin, J. Lacan, A. Bouabdallah, V. Roca, On-the-fly erasure coding for real-time video applications, IEEE Transactions on Multimedia 13 (4) (2011) 797–812.

[10] C.-H. Lin, C.-K. Shieh, N. Chilamkurti, C.-H. Ke, W.-S. Hwang, A RED-FEC Mechanism for Video Transmission Over WLANs, IEEE Transactions on Broadcasting 54 (3) (2008) 517–524.

[11] C. Lamoriniere, A. Nafaa, L. Murphy, Dynamic Switching between Adaptive FEC Protocols for Reliable Multi-Source Streaming, in: IEEE Global Telecommunications Conference, GLOBECOM, 2009, pp. 1–6.

[12] A. Sahai, Why Do Block Length and Delay Behave Differently if Feedback Is Present?, IEEE Transactions on Information Theory 54 (5) (2008) 1860–1886.

[13] P. Frossard, O. Verscheure, Joint source/FEC rate selection for quality-optimal MPEG-2 video delivery, IEEE Transactions on Image Processing (2001) 1815–1825.

[14] I. E. Richardson, The H.264 Advanced Video Compression Standard; electronic version, John Wiley & Sons Ltd, Chichester, 2010.

[15] S. Ma, W. Gao, Y. Lu, Rate-distortion analysis for H.264/AVC video coding and its application to rate control, IEEE Transactions on Circuits and Systems for Video Technology 15 (12) (2005) 1533–1544.

[16] Y. Jingyu, D. Qionghai, X. Wenli, D. Rong, A rate control algorithm for MPEG-2 to H.264 real-time transcoding, Visual Communications and Image Processing.

[17] x264 http://www.videolan.org/developers/x264.html.

[18] H.264/AVC reference software http://iphome.hhi.de/suehring/tml/.

[19] Tetrys website http://websites.isae.fr/tetrys.

[20] The Network Simulator http://www.isi.edu/nsnam/ns/index.html.

[21] ITU-T recommendation G.114, International Telecommunication Union, Tech. Rep. (2009).

[22] T. T. Tran, E. Lochin, J. Lacan, Online multipath convolutional coding for real-time transmission, in: 19th International Packet Video Workshop, 2012, pp. 41–46.

[23] S. Ahmad, R. Hamzaoui, M. Al-Akaidi, Unequal error protection using fountain codes with applications to video communication, IEEE Transactions on Multimedia 13 (1) (2011) 92–101.

[24] J. Sorensen, P. Popovski, J. Ostergaard, Feedback in LT Codes for Prioritized and Non-Prioritized Data, in: IEEE Vehicular Technology Conference (VTC Fall), 2012, pp. 1–5.

[25] S. Karande, H. Radha, Rate-constrained adaptive FEC for video over erasure channels with memory, in: International Conference on Image Processing (ICIP), 2004, pp. 2539–2542.

[26] P. Sadeghi, D. Traskov, R. Koetter, Adaptive network coding for broadcast channels, in: Workshop on Network Coding, Theory, and Applications (NetCod), 2009, pp. 80–85.

[27] L. Yang, Y. E. Sagduyu, J. H. Li, Adaptive network coding for scheduling real-time traffic with hard deadlines, in: Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 2012, pp. 105–114.

[28] T. Nguyen, A. Zakhor, Distributed Video Streaming with Forward Error Correction, in: IEEE Packet Video Workshop, 2002.

[29] ISO/IEC 23009-1:2012, Information technology Dynamic adaptive streaming over HTTP (DASH) Part 1: Media presentation description and segment formats.

[30] T. Stockhammer, Dynamic adaptive streaming over HTTP –: standards and design principles, in: Proceedings of the second annual ACM conference on Multimedia systems MMSys, 2011.

[31] D. Jurca, P. Frossard, Media flow rate allocation in multipath networks, IEEE Transactions on Multimedia 9 (2007) 1227–1240.

[32] Y. Zhang, W. Gao, Y. Lu, Q. Huang, D. Zhao, Joint Source-Channel Rate-Distortion Optimization for H.264 Video Coding Over Error-Prone Networks, IEEE Transactions on Multimedia 9 (2007) 445–454.

[33] Y.-W. Jeong, J. C. Kwon, J. kyoon Kim, K.-H. Park, A New Joint Source-Channel Modeling Approach for Adaptive FEC Code Rate Decision, in: IEEE International Conference on Multimedia and Expo, 2006, pp. 1433–1436.

[34] J.-W. Lee, C.-L. Chen, M.-F. Horng, Y.-H. Kuo, An efficient adaptive FEC algorithm for short-term quality control in wireless networks, in: 13th International Conference on Advanced Communication Technology (ICACT), 2011, pp. 1124–1129.

[35] A. L. H. Chow, H. Yang, C. H. Xia, M. Kim, Z. Liu, H. Lei, EMS: Encoded Multipath Streaming for Real-time Live Streaming Applications, in: IEEE International Conference on Network Protocols, Princeton, NJ, USA, 13-16 October, 2009, pp. 233–243.