The Journal of Systems and Software 109 (2015) 177–191

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

IP packet interleaving for UDP bursty losses ☆

Alessio Botta, Antonio Pescapé*

University of Napoli Federico II (Italy), Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione, Via Claudio, 21, Napoli, I 80125, NM2 srl, Italy

ARTICLE INFO

Article history:
Received 24 March 2015
Revised 14 July 2015
Accepted 30 July 2015
Available online 7 August 2015

Keywords:
Packet loss
UDP
Packet interleaving

ABSTRACT

The bursty nature of losses over the Internet is constantly asking for effective solutions. In this work, we use a comprehensive approach to study packet interleaving for coping with loss burstiness. Our aim is to assess if and how packet interleaving can be employed in real networks at the IP layer and what benefits real UDP-based applications may expect. Through an analytical study we determine the interleaving configuration to be used as a function of the loss characteristics. Then, in simulation we confirm these findings and highlight also potential counter effects. Afterwards, we design and develop a real application for packet interleaving. We move a step ahead towards real networks using a testbed that comprises an emulated WAN. Thanks to it, we study and solve a number of issues arising in real environments such as network dynamics and interleaving performance. Finally, we deploy the packet *interleaver* in the wild and we study the improvements achievable in general and by a real application. Our work shows that providing benefits to real applications is not an easy task. However, our packet *interleaver* can effectively decorrelate losses at the IP layer in real networks and highly increase real application performance, for example, video distortion can be reduced 65%.

1. Introduction and motivation

Packet loss, especially when happening in bursts, degrades the performance of Internet applications, affecting the quality perceived by the users (the so called Quality of Experience). Several studies in literature report the characteristics of the loss process in different environments (backbone (Yajnik et al., 1996), stub (Chung et al., 2004), and residential networks (Ellis et al., 2012; 2014; Mehmood et al., 2013)), with different kinds of traffic (unicast (Yajnik et al., 1999) and multicast (Bolot et al., 1995)), different protocols (UDP (Yajnik et al., 1999) and TCP (Paxson, 1999)), and at different time scales (sub-round trip time (Wei et al., 2007) or larger (Paxson, 1999)). They found, and we also experimentally confirm these findings (see Sections 6.2.1 and 6.3.1), that losses on the Internet are not isolated but rather happen in bursts. The problem is that the performance of the applications (e.g. streaming using predictive codecs such as MPEG) are more impacted by bursty losses than by the same quantity of isolated losses (Liang et al., 2003; Liu et al., 2009). This is true also for other classical UDP applications such as DNS (Ghita et al., 2010; Nguyen and Roughan, 2013; 2010; Wang et al., 2009). Recent studies evidenced that also classical TCP applications such as those based on FTP, especially when using some TCP versions, can be impacted by bursty losses (Wang et al., 2014).

To cope with this problem, different techniques have been proposed, which can roughly be classified as FEC- and ARQ-based. Such techniques normally imply overhead in terms of error-correction information, which have to be added before transmission (FEC) or retransmitted in case of loss (ARQ). *Interleaving*, also known as *time diversity*, represents a good candidate for network scenarios with bursty losses, allowing to spread the losses without transmitting additional information.

Different approaches have been proposed for applying interleaving in IP networks. As discussed in Section 1.1, they are typically based only on theoretical and simulative works, or, when presenting real implementations, they are tied to specific applications.

1.1. Related work

Diversity, in both time (Chin and Braun, 2001; Claypool and Zhu, 2003; Lee and Radha, 2005; 2006; Liang et al., 2003; 2002; Salvo Rossi et al., 2004) and space (Apostolopoulos and Trott, 2004; Bui et al., 2010; Hasegawa et al., 2005; He and Rexford, 2008; Nguyen et al., 2003; Tao and Guérin, 2004; Tao et al., 2004; Vergetis et al., 2006), represents a well known solution for coping with network scenarios affected by bursty losses. In this paper we focus on *packet interleaving* or *time diversity*. Here we provide a short review of the literature, for a more extensive discussion we refer the reader to (Botta, 2009).

☆ Very preliminary results within the same framework of this work have been published in A. Botta and A. Pescapé, "IP packet interleaving: Bridging the gap between theory and practice," in Computers and Communications (ISCC), 2011 IEEE Symposium on, 28 2011-july 1 2011, pp. 1022–1029.

* Correspondingauthor. Tel.: +390817683856.
E-mail addresses: a.botta@unina.it (A. Botta), pescape@unina.it (A. Pescapé).

http://dx.doi.org/10.1016/j.jss.2015.07.048
0164-1212/© 2015 Elsevier Inc. All rights reserved.

A simulation framework to study packet interleaving has been presented in literature (Salvo Rossi et al., 2004). Simulations have been performed using two channel models, one with uncorrelated random losses and the other one following a Markov chain. Delays have been assumed to be Gamma distributed, citing the work (Paxson, 1997), and a quality function to estimate the benefit of the packet interleaving has been introduced.

Interleaving has been often associated with videos. Liang et al. (2003) provided information about the effect of bursty rather than isolated losses on videos. They present a model to estimate the distortion caused by losses on videos. Simulation results show how a burst of losses actually causes more distortion than an equal number of isolated losses.

The same authors presented a study (Liang et al., 2002) on the effect of packet interleaving on real video sequences: a delay-distortion optimization problem is set, in order to choose the interleaving block sizes taking into account both the maximum acceptable delay and the minimal distortion objective; basically, knowing the maximum delay, they test all the possible configurations and choose the one providing the minimal average distortion according to the loss model.

On the other hand, Lee and Radha (2005); 2006) presented an interleaving scheme that operates before a generic video predictive encoder. Basically, the video stream is divided in two sub streams, which are then encoded separately. After encoding, the frames are rearranged in their original position. The decision on which frames to put into each sub stream is made using a Markov Decision Process, whose reward function takes into account the possibility that frames cannot be decoded because a reference frame has been lost.

Interleaving has also been employed to improve MPEG encoding. The system proposed in Claypool and Zhu (2003) interleaves the frames just before the encoder. The resulting sequence is more robust to possible frame loss but the compression is less efficient because some temporal redundancy is lost due to the reordering. The system has been evaluated using real videos ranked by real users. Results show that the average score is higher for interleaved frames even if a small bandwidth overhead is introduced.

MPEG encoder has also been considered to be more sensitive to isolated rather than bursty losses (Cai and Chen, 2001). Cai et al. explained that performing interleaving at symbol level before applying a channel coding scheme such as Reed-Solomon has the effect of protecting the bit stream from channel error, but results in isolated losses at the source (MPEG) level. Therefore they proposed to apply interleaving at source level (i.e. to the bit stream produced by the source encoder), then to apply the FEC, and then to re-apply the interleaving on the obtained symbols. This protects the stream from channel errors but results in bursty losses at the MPEG decoder. The results reported show that this scheme is able to increase the peak signal-to-noise ratio of about 5 dB.

Yao and Chen (1997) proposed a system that performs packet interleaving on MPEG audio, and they show the loss decorrelation power of their scheme both in simulation and on real networks.

Layered coding schemes are also particularly interesting for interleaving. Chin and Braun (2001) performed a simulation study of interleaving applied to such schemes. The proposed solution interleaves base and enhancement information. Simulations have been conducted by using loss patterns from real traffic, and they show how interleaving and randomization are able to actually protect the base information from bursty losses.

Random Linear Network Coding (RLNC) has also been studied with respect to interleaving and loss burstiness (De Alwis et al., 2012). This approach employs network coding together with interleaving and pre-coding in order to increase the robustness of h.264 video sequences. Results show that this new RLNC has higher performance than classical RLNC.

As for VoIP, Wah and Lin (1999) worked on the design and verification of a VoIP technique exploiting interleaving and a matrix-based transformation. The receiver continuously informs the sender about the status of the channel, i.e. the loss pattern. The sender then applies interleaving to the voice samples in order to compensate the effect of samples that will be lost.

VoIP has also been studied very recently by Soloducha and Raake (2014). The authors performed simulations to understand the effect of different kinds of losses (bursty and non) on VoIP encoded with recent codecs. The results show that loss burstiness has an important impact on the performance of VoIP with different codecs. The authors also suggested that loss burstiness is an important aspect to be considered in these kinds of studies.

On the same topic, Jelassi and Rubino (2011a); 2011b) performed a study of the accuracy of different methods to estimate the perceived quality of VoIP communications subjected to bursty losses. Results allow to understand how each assessor (i.e. estimator) captures the perceived quality at receiver side when the packets experience different levels of loss burstiness.

With regards to wireless networks, an interleaving scheme to alter the order of symbols before applying a FEC scheme has been proposed in Chen et al. (2004). Taking into account Bluetooth networks, the authors presented both analytical and simulation results that show how changing the order of the bits inside every single packets and then applying the standard Bluetooth FEC allows to obtain better performance. Simulation results related to TCP throughput are also presented.

Liu et al. (2014) also proposed an analytical framework and a metric for packet loss and its burstiness over wireless channels. The authors also proposed a packetization scheme that adaptively adjusts the packet size as a function of the predicted loss rate and burstiness.

1.2. Other techniques to cope with bursty losses

In this section we describe other techniques proposed in literature to cope with the bursty nature of losses. Being competitors of our proposal, we report and discuss such techniques to compare with them and to highlight the difference and improvements achievable with packet-level interleaving with respect to them. We provide information on this important aspect at the end of this section.

Cui et al. (2012) proposed to use network coding to cope with scarce bandwidth and losses over wireless multi-hop network for video streaming. The authors performed simulations to show the improvements achievable with their proposal. Results show that the video quality is highly improved especially in the presence of bursty losses.

In the same year, Casu et al. (2012) proposed to use a simplified FEC scheme to improve video quality at received side. The authors employed Low-Density Generator-Matrix (LDGM) codes applied to MPEG and showed, through simulations, that the percentage of packets recovered is higher with respect to other, more complex, FEC schemes in the presence of bursty losses.

New video codecs have also been studied recently (Oztas et al., 2012). In particular, the robustness of High Efficiency Video Coding (HEVC) has been studied using real loss traces. The traces used by the authors include bursty and non-bursty packet loss periods. Results indicate that HEVC is less robust to such kinds of losses with respect to H.264/AVC especially in scenes with high motion, even though the bandwidth required is smaller.

Studies on VoIP have recently been conducted with reference to new networking architectures such as Cognitive Packet Network (CPN), a Software Defined Network Substrate that provides user-oriented QoS by adaptively routing packets based on online sensing and measurement (Wang and Gelenbe, 2014). The authors show which metrics are to be considered in the QoS goal in order to obtain better overall performance for voice applications. Moreover, the authors show the causes of loss burstiness in this architecture and suggest how to cope with them.

On the same topic, authors of Nagano and Ito (2014) proposed a new coding approach, based on G.729, that is able to cope with large packet loss. The authors also used Gilber–Helliot model for losses, which takes into specific account loss correlation and therefore its burstiness. Authors showed which parameters of the codec have a major influence on the perceived quality of VoIP.

Han et al. (2012) proposed a new loss protection scheme called redundant packet transmission (RPT). According to the authors, the new scheme provides low latency with high robustness and is insensitive to parameter selection with respect to FEC. The scheme is validated through several analyses and with different kinds of losses that also comprise bursty ones. Results show that Redundant Transmission decreases the data loss rate by orders-of-magnitude more than typical FEC schemes applicable in live video communications.

Most of the works reported in literature either proposing the interleaving, or other techniques to cope with bursty losses, adopt a theoretical or simulative approach. The other (competing) techniques are also typically proposing solutions that imply communication overhead. A few works actually propose a real implementation and experimentation on real networks of packet interleaving. However, they adopt approaches that are application-specific, i.e. are not meant to work with other applications than the one targeted by the authors. Our aim is different. We aim at introducing a solution that is able to work with all the possible applications based on the Internet Protocol. The scientific literature has shown that several applications suffer from bursty losses and can therefore achieve better performance if the losses are decorrelated. For this reason, we do not want to target a single, specific application. Instead, what we believe is still missing in literature is a solution to decorrelate losses at IP layer, without looking at higher or lower layers, so to be independent of such layers. This is what we propose in this paper. In particular, in this paper we propose a more general solution and a more comprehensive approach for studying the applicability of interleaving at IP layer. Details on our contribution are reported in Section 1.4.

1.3. The case for interleaving at IP layer

In this paper we advocate the use of packet interleaving at the IP layer (also called packet layer or level in the following). As discussed in Section 1.1, the general idea of the interleaving is not new. Interleaving is used at MAC/Physical layer (e.g. in ADSL or on WiFi links) as well as at the application layer (e.g. together with specific video encoders). The benefits of those approaches have been extensively analyzed in literature and some of them are of large use today (i.e. in some ADSL configurations). However, these approaches have limited applicability with respect to ours, them being tied to a particular MAC/Physical layer technology or to a particular application layer protocol/architecture. Our approach is more general. IP is used today with almost all the possible lower and upper layer protocols, as the traditional IP hourglass shows. Therefore, the main novelty of our proposal comes from the use of the interleaver at the IP level, so to be independent of the underlying technology and the upper-layer application. For the same reason, we do not aim to compare the performance of packet-level interleavers with that of interleavers working at other layers, as we do not want to show that our proposal achieves higher performance than the others. We rather aim at showing that an IP layer with interleaving provides improved performance than standard IP layer, and we aim to convince the reader using a comprehensive approach, made of analytical, simulative, emulative, and experimental studies. The befits of decorrelated losses achievable by real applications have largely been demonstrated in literature. Being able to decorrelate losses at IP layer, without looking at the other layers, is therefore a way to achieve these benefits. In Section 6 we deeply analyze how loss decorrelation at IP layer can be achieved with our approach. In the same section we also show a case study of a real application using videos, achieving higher performance thanks to our approach.

1.4. Our contribution

In this paper we propose a comprehensive approach, made of analytical, simulative, emulative, and experimental studies, for studying the applicability of time diversity at IP level in real networks affected by bursty losses, and we show how the proposed interleaver can profitably be used in real networks without being dependent on a particular application, codec, or protocol. Moreover, we present, for the first time in literature, a real application for packet interleaving and we analyze, through theoretical, simulative, emulative, and experimental studies, several issues for the deployment of such application in real networks.

In this work we explicitly target UDP flows. Our choice is motivated by the expected rise of UDP traffic volume (Dang et al., 2006; Finamore et al., 2010; Lee et al., 2012; Zhang et al., 2009), which steps from the momentum of applications (e.g. DNS, VoIP, streaming, p2p IPTV, etc.) and new protocols (e.g. uTP) for p2p applications that deeply rely on UDP, also to avoid traffic shaping techniques. As reported in Zhang et al. (2009), in the period 2002-2009, UDP has gained popularity, especially considering the number of flows: UDP allows efficient establishment and maintenance of p2p overlay networks, while the use of random ports evades detection by port-based traffic engineering or filtering techniques. This increasing trend in UDP usage has been seen all over the world, with a peak in data from China where UDP-based p2p IPTV traffic is already common. Finally, recent studies confirm this trend: in Lee et al. (2012) the authors have observed a relevant increase in terms of UDP traffic volume. More precisely, by continuously monitoring the same link for four years they found a 46-fold increase in volume (from 0.47 to 22.0% of total bytes). Finally, UDP is the protocol adopted by the most used applications for audio/video communication, e.g. Skype. Skype uses both TCP and UDP: TCP for control messages and UDP for video transmission and the sender adapts its UDP sending rate to network conditions (Zhang et al., 2012). Also, as for the other UDP-based applications, it is worth noticing that in the case of DNS in Ghita et al. (2010); Nguyen and Roughan (2013); 2010); Wang et al. (2009) the authors found evidences that most losses occur in bursts: they often found long periods of no-loss, followed by short bursts of losses.

Summarizing, our work extends the literature in that: (i) we propose a comprehensive approach, made of analytical, simulative, emulative, and experimental studies, for analyzing, and deploying time diversity at IP level in real networks affected by bursty losses; (ii) we analytically derive the best interleaving configurations as a function of the network losses (Section 2); (iii) we confirm the obtained results in simulation, evidencing also possible counter effects (i.e. isolated losses may become very close to each other) (Section 3); (iv) we present a tool, called *TimeD*, implementing time diversity at packet level (Section 4.1); (v) we analyze, discuss, propose, and integrate in *TimeD* effective solutions for several issues arising from the use of an interleaver in real networks (i.e. block size to be used as a function of the loss patterns, estimation of the network status, packet size and packet rate of the probing traffic, buffering timeout configuration, packet release strategy, performance overhead introduced by the interleaver) (Section 4.2); (vi) we validate the tool in an emulated environment (Section 5); (vii) we study the burstiness of losses over real wired (PlanetLab) and wireless (Satellite) networks providing evidences and quantitative evaluations (Section 6); (viii) we provide results on the use of *TimeD* in real networks, demonstrating its positive impact on loss burstiness and performance improvement for general IP-based applications and for an example UDP-based video application (Section 6); (ix) we publicly release the implemented tools, both the simulator and *TimeD*.

In our work we study the improvements achievable by real applications thanks to several advanced features of *TimeD*. For example, if it is deployed on a satellite home router, *TimeD* improves the performance of UDP-based video streaming. We measured, in fact, that *TimeD* reduces the video distortion of more than 65% for a streaming of a video with 256-bytes packets over a downstream satellite channel (see Section 6). Also, even if it is out of the scope of this paper, other classes of UDP-based applications benefiting the use of *TimeD* are DNS and file sharing applications based on uTP.

2. Analytical study

In this section we study the problem of packet interleaving from an analytical point of view. We firstly illustrate the loss model we use and the type of interleaving we choose, and then we show how to analytically determine the interleaving configuration as a function of the losses.

2.1. Loss model

We assume that the loss process we are dealing with follows a 2-state Markov chain (2-MC) (Ribeiro et al., 2005; Wei et al., 2007). Such model is used for the analytical studies (presented in the following of this section) and for the analyses performed both in simulation (see Section 3) and emulation (see Section 5). In Sections 6.2.1 and 6.3.1 we also evaluate the characteristics of the loss process respectively over Planetlab and a satellite network. The loss model based on 2-MC is also known as Gilbert–Elliott model (Elliott, 1963; Gilbert et al., 1960). Such model has been shown to be able to capture the potential correlation between consecutive losses both on the Internet (Yajnik et al., 1999) and on wireless networks (Chen et al., 2004). It is also worth noting that 2-MC is not the only or the fittest model for packet loss (Jelassi and Rubino, 2013). It has actually been shown that Markov chains with more states are able to obtain higher modeling accuracy in some cases (Yajnik et al., 1999). However, 2-MC represents the best compromise between complexity and accuracy.

Let $X = \{X(t) : t \geq 0\}$ be the random process of losses following the Gilbert–Elliott model. The state X at time t can assume one of the following values: b or g (b = "bad" and g = "good"). The process $X(t)$ at a fixed time is characterized by the parameters μ_g and μ_b, which can respectively be seen as the rates at which the chain passes from state g to state b and vice versa. In general, when $X(t)$ is in state b the probability to lose a packet is much larger than that in state g. To simplify the problem, we assume that the former probability is 1 and the latter is 0. For this reason we refer to the two states as "*loss*" and "*no-loss*", besides "*bad*" and "*good*" respectively. The steady state probabilities to receive or lose a packet are respectively equal to: $\omega_b = \frac{\mu_b}{\mu_b+\mu_g}$ and $\omega_g = \frac{\mu_g}{\mu_b+\mu_g}$. It is worth noting that we consider a packet to be lost also when it is delivered too late, and it is therefore useless for the application (e.g. audio samples arriving after playback time). As we are only interested in the behavior of the network when packets are sent, we substitute the continuous-time model with a discrete-time one in which the discrete time represents the departure time of the packets. Once the sending rate is fixed to $S = 1/\tau$, with τ being the inter-packet time[2], we can express the transition probabilities in the discrete case as:

$$p_{gg}(\tau) \equiv P(X_{n+1} = g|X_n = g) = \omega_g + \omega_b e^{-(\mu_g+\mu_b)\tau} \quad (1)$$

$$p_{gb}(\tau) \equiv P(X_{n+1} = g|X_n = b) = 1 - p_{gg}(\tau) \quad (2)$$

$$p_{bb}(\tau) \equiv P(X_{n+1} = b|X_n = b) = \omega_b + \omega_g e^{-(\mu_g+\mu_b)\tau} \quad (3)$$

[2] Here we are assuming CBR traffic. The model derivation can be extended for other traffic types (see Botta, 2009 for more details). Interleaving results with VBR traffic are reported in Section 6.

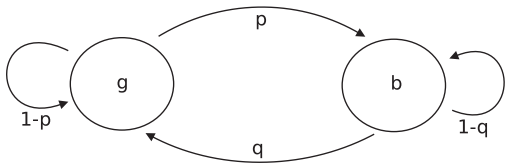


Fig. 1. 2-state Markov chain.

$$p_{bg}(\tau) \equiv P(X_{n+1} = b|X_n = g) = 1 - p_{bb}(\tau) \quad (4)$$

Assuming the homogeneity of the Markov chain and choosing $\tau = 1$, the probabilistic description of such process can be obtained by using a 2-MC (see Fig. 1) having $p = p_{bg}(1)$ (probability that the next packet is lost given that the previous is not) and $q = p_{gb}(1)$ (probability that the next packet is received given that the previous is lost).

In general, the following condition holds: $p + q \leq 1$. If $p + q = 1$ the model becomes a Bernoulli one, for which losses are independent. The steady-state probabilities for the two states b and g are π_b and π_g respectively:

$$\pi_b = \frac{p}{p+q}, \quad \pi_g = \frac{q}{p+q} \quad (5)$$

π_b is the first important parameter of the model because it represents the average loss probability. The second important parameter is ρ, which represents the *temporal correlation between losses*, defined as follows:

$$\rho \equiv \frac{1-q}{p} \quad (6)$$

To obtain a Bernoulli model it is sufficient to impose $\rho = 1$, and the losses will be independent. This implies that a loss has the same probability to happen given that the previous state was b or g. On the other hand, $\rho > 1$ implies that a loss is more probable if the previous state was b, i.e. if the previous packet was lost. For this reason, ρ is considered as an indicator of the channel memory. The last important result is related to the probability of transition from state i to state j in l steps:

$$p_{ji}(l) \equiv P(X_{n+l} = j|X_n = i) \quad (7)$$

Starting from the 1-step transition matrix:

$$\mathbf{Q} = \begin{pmatrix} (1-p) & p \\ q & (1-q) \end{pmatrix}$$

We have to calculate the l-step transition matrix $Q_l = Q^l$, and recall that (Rodney, 1974):

$$p_{bg}(l) = Q_l(1, 2)$$
$$p_{gb}(l) = Q_l(2, 1) \quad (8)$$

Otherwise we can directly use the following (Vergetis et al., 2005):

$$p_{bg}(l) = \frac{p}{q+p}[1 - (1-q-p)^l]$$
$$p_{gb}(l) = \frac{q}{q+p}[1 - (1-q-p)^l] \quad (9)$$

2.2. Block interleaving

The basic idea to realize packet interleaving is to resequence packets before transmission. Resequencing allows to space out originally close packets, so that a loss of consecutive packets is translated in a loss of distant ones. The main drawback of packet interleaving is the delay introduced. In order to resequence a certain number of packets, it is necessary to wait for them. Such delay is however acceptable for several applications (e.g. audio and video streaming, p2p file sharing, etc.) and, more importantly, it is controllable through the length of the sequence.

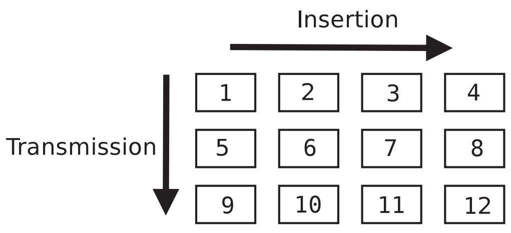


Fig. 2. Block interleaving.

When the length of the sequence is chosen, determining the optimal permutation of such sequence is not an easy task. As exhaustive searching the optimal permutation is usually not feasible, interleaving is commonly implemented in blocks (we refer to it as to *block interleaving*). Block interleaving is made inserting packets into a matrix by rows (Insertion), and picking them by columns (Transmission), as shown in Fig. 2. Basically, the traffic flow is divided in sequences of k packets. Each sequence is then placed into a *block*: a matrix of size $k = n \times m$ where n, the number of rows, is called *interleaving depth*.

The effectiveness of such transmission schema depends on the values of n and m, as well as on network conditions[3]: a burst of length B will be converted in smaller bursts of length B/n. When $n \geq B$ we are able to convert the burst in an equivalent number of isolated losses, spaced of m or $m-1$ packets. Therefore, increasing n and m, the capacity of converting bursts into isolated losses increases. On the other hand, there are two counter effects: (i) the number of smaller bursts increases; (ii) the buffering delay increases.

2.3. Block dimensioning

As reported before, using a block interleaver with interleaving depth of n, it is possible to transform a burst of n packets into single losses separated by m or $m-1$ packets. This provides a first piece of information in order to properly dimension the block size. In fact, if the loss bursts had always a fixed size (e.g. 3 packets), we could easily dimension the interleaving block so to decorrelate all such bursts (i.e. $n \geq 3$). As the bursts have different sizes and we want to choose the minimum possible block size, we have to find a probabilistic criterion. Of course, a first criterion could be that of setting the block size equal to the average burst length. This would allow to decorrelate all the bursts that have a length less than or equal to the average length. To apply this criterion we have to find from the Gilbert–Elliot model such average burst length, which is equal to the expected sojourn time in state b (see Fig. 1):

$$E(burst\_length) = \frac{1}{q} \tag{10}$$

and then impose:

$$n \geq \frac{1}{q} \tag{11}$$

A possible issue with this criterion is that, as we will see in the following, the loss-burst length distribution may have a heavy right tail (i.e. be right skewed), meaning that few samples will be smaller than or equal to the average value. Therefore, with this criterion we decorrelate a small percentage of the loss bursts. A more effective criterion could be that of configuring the interleaver so to decorrelate a determined fraction of the loss-bursts. In more detail, let us say that we want to decorrelate the d% of all the loss bursts. Then, we have to set the interleaving depth n to be equal to the dth-percentile of the distribution of the burst length:

$$n \geq dth - percentile \tag{12}$$

We call this interleaving depth the *d%-depth*. So using (12) we are sure to decorrelate the d% of all the loss bursts on the channel.

[3] Once k and n are chosen, m is automatically determined.

This criterion is better than setting n equal to the average loss-burst length mainly because we have a parameter (d) that allows to control how effective we want the interleaver to be. Clearly, given a certain channel, the more we increase d, the more we increase n, and consequently the buffering delay. Therefore, we have to find the proper trade-off, considering the application scenario.

To apply (12) we have to determine the dth-percentile of the loss-burst length distribution. We do this in the following. The probability to have a loss burst of 2 packets is equal to the probability of moving from state g to state b, then staying in state b, and then moving back to state g, given that we started in state g. This can be calculated as:

$$\begin{aligned}
&P(burst\_of\_length\_2)\\
&\quad = P(X_{n+3} = g, X_{n+2} = b, X_{n+1} = b | X_n = g)\\
&\quad = P(X_{n+3} = g | X_{n+2} = b) P(X_{n+2} = b | X_{n+1} = b)\\
&\qquad P(Xn + 1 = b | X_n = g)\\
&\quad = p(1-q)q
\end{aligned}$$

Generalizing, the probability to have a loss burst of length z packets, can be obtained as follows:

$$\begin{aligned}
&P(burst\_of\_length\_z)\\
&\quad = P(X_{n+z+1} = g, X_{n+z} = b, \ldots, X_{n+1} = b | X_n = g)\\
&\quad = p(1-q)^{z-1} q
\end{aligned} \tag{13}$$

Being interested only in the burst length and not in the probability of having a burst, from the previous we obtain the probability that the loss burst will have a length of z packets (i.e. the relative frequency of that particular loss-burst length with respect to the other loss-burst lengths):

$$P(burst\_length = z) = (1-q)^{z-1} q \tag{14}$$

From (14) we can calculate the cumulative distribution function (CDF) of the loss-burst length as:

$$\begin{aligned}
F(x) &= P(burst\_length < x)\\
&= \sum_{i=1}^{x} (1-q)^{i-1} q\\
&= q \sum_{i=1}^{x} (1-q)^{i-1}
\end{aligned} \tag{15}$$

Finally, using (15), we can find the dth-percentile and, therefore, the *d%-depth* by finding the $\hat{x}$ in the following:

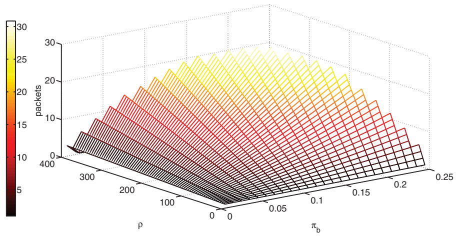
$$F(\hat{x}) = q \sum_{i=1}^{\hat{x}} (1-q)^{i-1} = \frac{d}{100} \tag{16}$$

Summarizing, if we want to decorrelate the d% of the loss bursts, we have to find the $\hat{x}$ from (16), and then set interleaving depth n as:

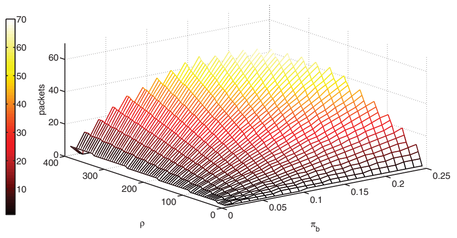
$$n \geq \hat{x} \tag{17}$$

In the following we show an example of the application of this criterion. In Fig. 3(a) we report the average loss-burst length for channels having losses following a 2-MC with values of $\rho \in [0, 400]$ and of $\pi_b \in [0, 0.25]$. It has been calculated inverting (5) and (6) and using (10). In Fig. 3(b) we report the 90th-percentile and in Fig. 3(c) the 75th-percentile of the loss-burst distributions for the same channels (i.e. same values of ρ and π_b). They have been calculated using (16). We decided to consider these ranges of ρ and π_b because they are consistent with the values obtained in real IP networks (see Section 6). It is also worth noting that not all the couples of ρ and π_b are possible in practice (i.e. when π_b is high, ρ cannot be high, and vice versa). For this reason, the three plots of Fig. 3 reports only the values related to the possible values of ρ and π_b.

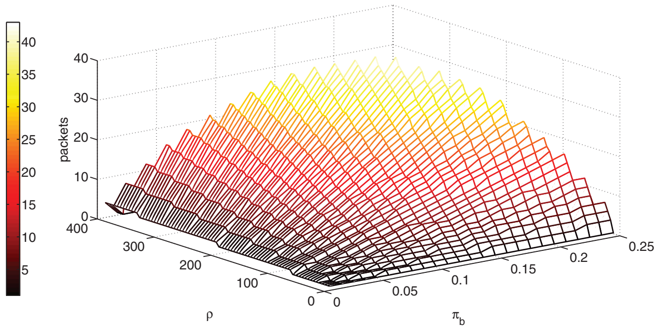
If we want to decorrelate the 90% of the loss bursts, the minimum necessary interleaving depth (i.e. the 90%-*depth*) is equal to values



(a) Average loss-burst length.



(b) 90th-percentile of the loss-burst length.



(c) 75th-percentile of the loss-burst length.

Fig. 3. Average, 90th-percentile, and 75*th*-percentile of the loss-burst length with different values of ρ and π.

reported on the *z*-axis of Fig. 3(b). As shown, in the worst channel conditions, the 90%-*depth* of the interleaver reaches values up to 70 packets. This means that the interleaver block has to be at least of 140 packets (for a minimum size block, we only use 2 columns), and we have to buffer such amount of packets for every block. Such a buffering delay may be not acceptable for some applications. Looking at the *z*-axis of Fig. 3(c) we can see the 75%-*depth*. In this case, we observe a maximum interleaving depth of about 35 packets, which is obtained in the worst channel conditions (i.e. when the average loss-burst length is about 30 packets). Using the 75%-*depth* instead of the 90%-*depth*, we can highly decrease the buffering delay (i.e. about half of the time), accepting to decorrelate only the 75% of all the loss bursts.

The two examples reported above show how the final choice of which interleaving depth to use (the average, the 90%-, the 75%-*depth*, or others) depends on the kind of application we are dealing with. For example, the analytical analysis allowed to see that an interleaving depth of 35 packets is sufficient to decorrelate 75% of all the bursts happening even in the worst channel conditions, where we have an average loss burst length of 30 packets. The criteria proposed in this paper allows to easily calculate the different parameters of the interleaving block, therefore making informed decisions on how to configure an interleaver for a real network. In the following we will see how these criteria have been used in our real application for packet interleaving.

3. Simulation study

The second important step of our analysis is then performed in simulation. We implemented a simulator (using Matlab^{TM}) to reproduce the behavior of a block interleaver operating on a lossy channel. Thanks to our simulator, we evaluate the distribution of the length of the loss bursts and the distribution of the distance between two loss events (we call this distance *no-loss sequence length*). These parameters are evaluated in general and with different interleaving configurations. This analysis allows us to understand the benefits achievable with the block interleaver and the possible counter effects. It is worth noting that once the average loss rate is fixed, the loss bursts and the no-loss sequences are tightly linked: the more we reduce the loss-burst length, the more we bring losses closer to each other. For this reason, the no-loss sequence length may seem a redundant parameter. However, in some cases, too close loss events may cause effects similar to loss bursts, decreasing application performance (e.g. video applications using H.264/AVC codecs, when the distance between losses becomes smaller than a threshold (Liang et al., 2003)). Due to space constraints we briefly discuss here the most important simulation results. More details are reported in Botta (2009). An alpha version of the simulator is publicly available at http://traffic.comics.unina.it under the GNU General Public Licence terms.

Table 1
Parameters used in simulation.

Parameter	Values
Interleaving block size	[12, 24, 48]
Interleaving block depth	[1, 2, 6, 12, 24]
π_b	[0.01, 0.03, 0.1, 0.25]
ρ	[1, 3, 8, 15, 30]
Repetitions	1000

To understand the impact of each single parameter, we performed simulations using all the possible combinations of the values reported in Table 1. In the following we report and discuss the results that are more useful to assess the impact of the interleaving with different channel conditions. Moreover, we present and discuss results in terms of loss-burst length because this parameter is experimentally measured and discussed later on. Fig. 4 shows the average values of the loss-burst lengths (4(a) and (c)) and no-loss sequence lengths (4(b) and (d)) as a function of the interleaving depth, when using a block size of 48. Unless explicitly reported, the considerations made in the following apply also to the other block sizes. Fig. 4(a) and (b) are related to four representative combinations of ρ and π_b, with

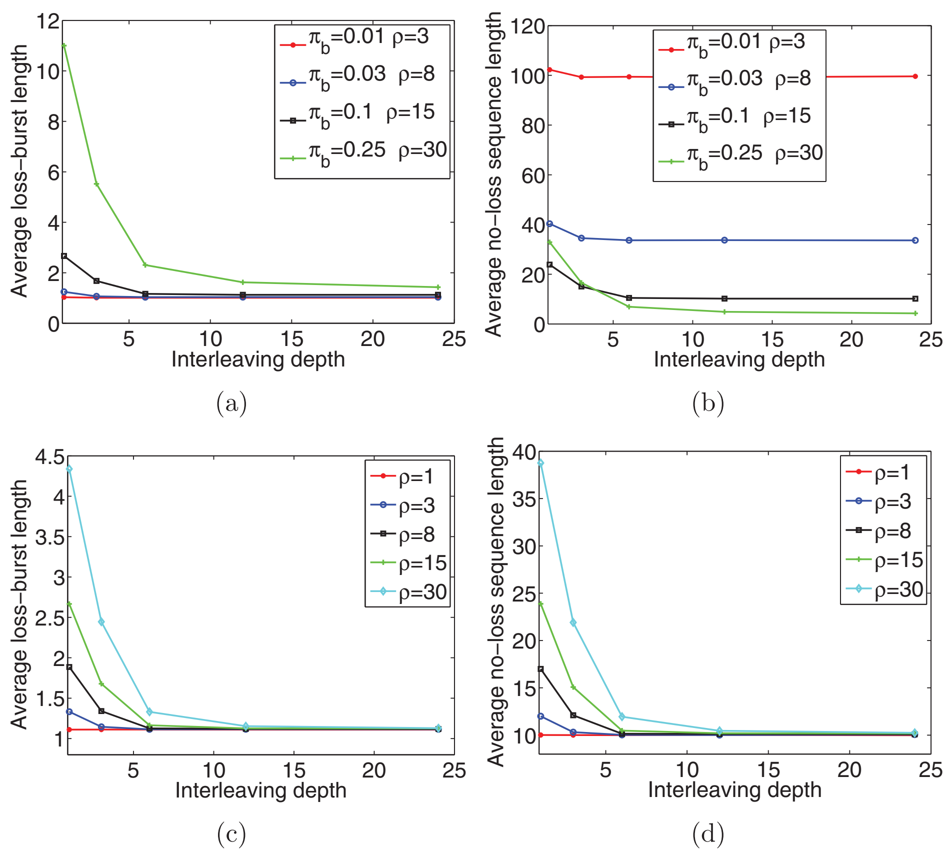


Fig. 4. Simulation results obtained with different values of π_b and ρ.

progressively degraded channel conditions. Fig. 4(c) and (d) are obtained using $\pi_b = 0.1$ and increasing ρ. As a first consideration, the results allow to understand the effectiveness of the interleaving in reducing the average loss-burst length (i.e. the loss decorrelation power). As shown in Fig. 4(a) and (c), in all the cases such parameter presents a decreasing trend when increasing the interleaving depth. We also observe that the curves are steeper when the length of the loss bursts is high, i.e. the slope decreases when the interleaving depth increases. The same behavior can be observed comparing the different plots in Fig. 4(c): the higher the correlation, the more effective the interleaving. This translates in an asymptotic behavior of the curves, which tend to a line parallel to the x-axis, corresponding to a memoryless channel. This happens because after a certain depth, that depends on the channel conditions, the interleaving manages to make the losses perceived as uncorrelated, and the channel behaves as the Bernoulli one. Further increasing the interleaving depth provides no benefit in terms of average loss-burst length. However, we know from the analytical study that the interleaving depth may be pushed further to decorrelate the 75% or the 90% of all the loss bursts.

This behavior is also confirmed by the plots of the no-loss sequence length (Fig. 4(b) and (d)). When the block size is fixed, pushing the interleaving depth (n) m decreases, and consequently the length of the no-loss sequences decreases and losses become closer to each other. The plot shows also that the length of the no-loss sequences reaches the value of the Bernoulli channel, which represents an asymptote also for this parameter. Also, we observe that the trend of the plots is very close to that of the loss-burst length (especially visible in the right plots). This is due to the fact that the interleaving does not alter the average loss rate, which can be roughly seen as the ratio between the average loss-burst length and the sum of the average no-loss sequence length and the average loss-burst length.

This analysis allowed to assess the potential benefits and harms of packet-level interleaving on a network with losses following a 2-MC. The obtained results will also be used as a reference for the validation of our application for packet interleaving, which is presented in the next section.

4. TimeD: time diversity at packet level

We implemented time diversity at packet level, using *block interleaving*, in a platform we called *TimeD*, which is described in the following.

4.1. TimeD *design and implementation*

TimeD is a user-space application that interleaves IP packets generated by or traversing a Linux host. It is written in C language and works over Linux platforms. Besides other interesting features, Linux provides more flexibility and allows the interleaving application to be easily deployed as an embedded system in any operational network. *TimeD* is a user-space application, which allows to interact with user-space monitoring applications, whose output is useful to tune the interleaving parameters (as we will see in the following). To modify the order of the packets as required by time diversity, *TimeD* uses the *libipq*[4]: a mechanism provided by *netfilter*[5] for passing packets out of the kernel stack, queuing them to user-space, and receiving them back into the kernel with a verdict specifying what to do (e.g. ACCEPT or DROP). Thanks to the use of *libipq*, *TimeD* is flexible and its use can be customized according to the reference scenarios. As *TimeD* is

[4] https://svn.netfilter.org/netfilter/trunk/iptables/libipq
[5] http://www.netfilter.org

based on *iptables*, the packets to be interleaved can be selected using several criteria such as the destination host, transport protocol and ports, etc.

Designing and implementing *TimeD* we had to change its architecture and operating mode different times to cope with real traffic and networks. We report here the main lessons learned and the final design of the tool. At a very high level, *TimeD* operates in a cyclic fashion. During each iteration, it performs the following operations: (i) if present, read a packet from the input queue and put it in a block-sized buffer; (ii) if the buffer is full or a timeout has expired, evaluate some statistics, interleave the packets in the buffer, and put them in the output queue; (iii) if present, release a packet from the output queue with a particular strategy. Moreover, an additional module is activated periodically to estimate the status of the network and configure the interleaving block accordingly. These operations have been carefully studied and designed before the development of *TimeD*. In fact, big issues in moving packet interleaving from theory to practice lay in how these operations are carried out. In the following we provide details about them. In particular, in Section 4.2.1 we detail the way we configure the block size, and report which statistics are calculated and how. In Section 4.2.2 we describe how the timeout is calculated, and the strategy purposely designed for releasing the packets in the output queue is detailed in Section 4.2.3.

It is worth noting that implementing an interleaver that works at the application layer (e.g. inside a video application) may be easier than implementing one at the network/IP layer. In the former case the application has full control over the packets/messages transmitted and received, having also direct visibility on what happens to these packets (delivery or not, delay, etc.). Most of the issues discussed in the next sections may therefore be less complex for an application-layer interleaver. However, such an interleaver would be tied to that particular application, losing the generality we are targeting at with IP-layer interleaving.

An alpha version of *TimeD* is publicly available at http://www.grid.unina.it/Traffic under the terms of GNU General Public License.

4.2. *Most important operating features*

We considered several issues in order to deploy packet-level interleaving in real IP networks.

4.2.1. *Block size*

From what we showed, it should be clear to the reader that one of the most important issues of block interleaving is related to the fact that the block size has to be tuned according to the loss pattern on the network. And, in order to avoid unnecessarily large buffering delay, we have always to choose the smallest possible block size (mainly in terms of interleaving depth) that provides the required loss decorrelation. To this end, we integrated an active probing tool in *TimeD*. Thanks to it, *TimeD* periodically performs active measurements, estimates the loss pattern on the network in terms of π_b and ρ (see Section 2.1) from the received traffic, and adjusts the block size accordingly. In order to effectively perform these tasks, several important aspects have to be considered and deeply investigated. The most important ones are reported in the following.

- *The block size to be used as a function of the loss pattern: TimeD* sets the size of the block according to the results of the analytical study reported in Section 2.3. We use the 75%-*depth* because, in general, it provides the best trade-off between loss decorrelation and buffering delay. A different policy (average, 90%-*depth*, etc.) can also be used when targeting a particular application.
- *The packet size and rate of the probing traffic*: to choose these parameters, we performed a large set of experiments in an emulated scenario. We decided to perform this analysis in emulation because, in such conditions, we can use real operating systems, applications, and traffic on top of an emulated network, for which we can completely control the loss behavior. This allows on the one side, to use the real implementation of the tool, and therefore to also validate it, and on the other side, to have a reliable reference value for the variables to be estimated. In Section 5 we present the emulation environment and the main results obtained.
- *How frequently to estimate the status of the network (i.e. the measurement period)*: the measurement period is in strict relation with how frequently the loss pattern on the network changes. To tune this parameter we both studied the literature and performed a set of experiments over real networks, as described in Section 6.3.1.

4.2.2. *Buffering timeout configuration*

TimeD buffers incoming packets until the block is full and the interleaving can take place. In order to limit the maximum delay that a packet can experiment, we introduced a timeout mechanism for the buffering operation: the packets are released if the timeout fires. The value of the timeout, can be either specified by the user or autonomously determined by *TimeD*. To achieve this, *TimeD* estimates the average rate of the received packets, and then sets the buffering timeout as a function of this rate. The idea is that, knowing the rate and the block size, we can reasonably predict the time needed to fill the block. The estimation of the application rate is also used to release the packets with the same rate they have been received by *TimeD*, as explained in the next section. To estimate the rate, *TimeD* works as follows. Let us suppose that we want to operate with a block of size $n \times m = k$, on a flow of packets $pkt_1, pkt_2, \ldots, pkt_a, \ldots$, whose arrival times are $t_1, t_2, \ldots, t_a, \ldots$. Every time we release the current block, because either the block is full or the timeout (TO) is expired after receiving c packets (with $c < k$), we calculate a new rate sample $r(j)$:

$$r(j) = \frac{k}{t_a - t_{a-k}} (\text{if block is filled})$$

$$r(j) = \frac{c}{TO} (\text{if timeout is expired})$$

We calculate the average and standard deviation of the flow rate using the last α rate samples[6]:

$$r_{avg} = \frac{1}{\alpha} \sum_{i=0}^{\alpha-1} r(j-i); \quad r_{std} = \sqrt{\frac{1}{\alpha} \sum_{i=0}^{\alpha-1} (r(j-i) - r_{avg})^2} \tag{18}$$

And, we set the timeout as follows:

$$TO = \frac{r_{avg} + 4 * r_{std}}{k} \tag{19}$$

4.2.3. *Packet release strategy*

Once the packets are buffered and interleaved, *TimeD* releases them with the same rate they have been received. This is to preserve as much as possible the original profile of the traffic, and to avoid artificial spikes due, for example, to sudden release of all the packets. Besides distorting the application rate, such sudden release can also introduce higher packet loss, as we experimentally verified on satellite networks.

As explained in the previous section, *TimeD* estimates the rate of the packets received in every block ($r(j)$). This value, is stored together with the packets in the output queue. Using this value, *TimeD* performs the following operations continuously: (i) pull the next packet from the output queue and release it; (ii) wait for a time equal to $1/r(j)$, while performing the operations related to the incoming packets (i.e. check if packets are present in the input queue, move these packets to the buffer, evaluate the statistics, and put the packets in

[6] We empirically found that a value of α equal to 8 is enough for our aims.

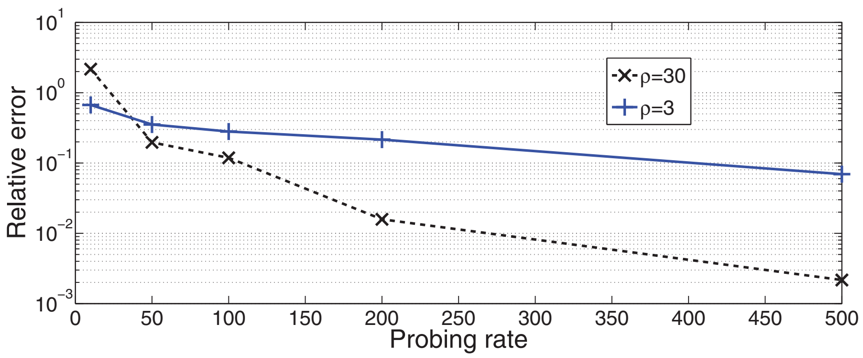


Fig. 5. Relative error of loss correlation estimation.

the output queue if the buffer is full or the timeout is expired); (iii) verify if the time elapsed is larger than the expected. This last condition is extremely important. The receiving-related operations can sometimes take longer than expected (e.g. when the process is descheduled from the cpu). This would cause an additional delay in releasing the packets. In some cases, such as with high-rate CBR traffic, this can also imply a monotonically increasing queue (i.e. packets waiting for longer and longer). To avoid that, we continuously monitor the elapsed time, and, in case we are in late, we recover the delay during the next cycles (i.e. we release more packets at once). The result is that some packets can leave the buffer before their scheduled time. However, they compensate for those which left the buffer in late, with the final result that the original rate is respected in average, and no blocks are buffered more than necessarily.

4.2.4. *Performance*

Passing packets from kernel- to user-space impacts the performance and poses a limit on the maximum rate supported by *TimeD*. We verify that the delay introduced by *TimeD* is negligible with respect to the one of our experimentation scenarios (Section 5.2.1), and that the tool was able to sustain the rate considered in the experiments (Section 5.2.2). We believe that performance issues have to be carefully taken into account before the deployment of *TimeD* in operational environments. We left this as a future work, in which we also consider whether it would be beneficial (and at which cost) to move packet interleaving to the kernel space.

5. Emulation study

The next step of our analysis is performed in emulation. According to the discussion reported in Section 4.2.1, we use an emulative approach for validating *TimeD* and checking its implementation over (controlled) real networks. In order to validate *TimeD* behavior, we firstly analyze if and how its probing capabilities allow *TimeD* to effectively estimate the loss characteristics. Secondly, we evaluate the performance of *TimeD* in terms of additional delay introduced and achievable throughput. Thirdly, we verify its loss decorrelation capabilities in time-varying conditions.

The testbed we use is composed of Linux hosts connected (using a Fast Ethernet network) to the Shunra Virtual Enterprise WAN emulator[7]. Such hardware emulator is able to reproduce the behavior of a WAN in a controlled manner, allowing to set different parameters. For our experiments, we set a loss pattern equal to a 2-MC, and leave the delay and jitter unset (i.e. no delay or jitter is intentionally added). For generating probe traffic we use D-ITG (Botta et al., 2012), which emulates a real network application exploiting the benefits of packet interleaving. Before running the validation of *TimeD*, we verified its ability to correctly decorrelate losses: we observed that the average values of the length of the loss bursts and no-loss sequences was decreasing with the increase of interleaving depth, as in simulation. Details about the configurations we consider in our experiments are reported in Table 2. Further details and complete results of this stage are reported in Botta (2009).

Table 2
Parameters used in emulation.

Parameter	Values
Interleaving block size	[12, 24, 48]
Interleaving block depth	[1, 2, 3, 6, 12, 24]
π_b (*loss*)	[0, 0.01, 0.03, 0.1, 0.25]
ρ (*rho*)	[0, 1, 3, 8, 15, 30]
Repetitions	20
Protocols	UDP
Average packet rate	10, 100, 1000 pps
Average packet size	64, 128, 256, 512, 1024, 1472 bytes

5.1. *Probing flow characteristics*

TimeD performs active probing on the network to estimate the loss characteristics and set the proper block size. Therefore, it is important to understand how to perform such probing, in order to obtain a sufficiently accurate estimate for this aim. Different contrasting interests have to be considered, that are *measurement intrusiveness* and *accuracy*. In fact, the more accurate we want our measurements to be, the more samples we have to collect. And, once the measurement period is fixed, to collect more samples we have to probe the network with a higher packet rate, which implies a higher intrusiveness. To choose the parameters of the probing flow, we have performed a large set of measurements, varying the probing traffic profile (packet rate and size) and the loss pattern on the emulated WAN. We then looked at the values of loss rate and correlation estimated by *TimeD*. The results show that the loss rate is easily estimated with a small probing rate: a probing rate of 10 pps is already enough to obtain a relative error in the order of 10^{-1}, which is sufficient for our aim.

For the rate correlation, we observe a different situation. Fig. 5 shows the relative error obtained in two experiments performed using $\pi_b = 0.01$ and $\rho \in \{3; 30\}$. As shown, at low probing rates (e.g. 10 pps) the maximum relative error value is about 2×10^0. However, the results of the simulations performed in Section 3 indicate that it is necessary to estimate the loss correlation with an absolute error in this order of magnitude (i.e. 10^0). It is also worth saying that, in order to observe the same channel conditions experimented by the application, it is not necessary to probe the network with a rate higher than that of the application. For these reasons, we choose to probe the network at a packet rate equal to the minimum between that of the application and 50 pps. This guarantees both a sufficient accuracy and a low intrusiveness of the probing traffic (50 pps × 50 bytes/pkt = 20 Kbps).

[7] http://www.shunra.com/shunra-ve-overview.php

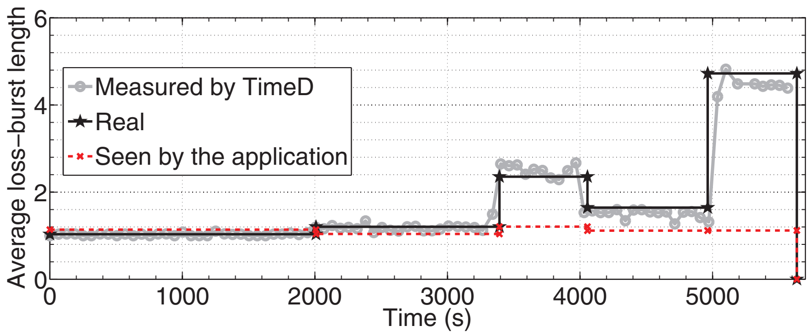


Fig. 6. Results obtained with *TimeD* with time-varying channel conditions.

5.2. TimeD *performance*

5.2.1. *Additional delay introduced*

In simulation we neglected the delay issues because we wanted to understand the potential benefit of packet interleaving. However, if we want to deploy the interleaving on a real network we have to consider all the effects that we might expect. For these experiments we connect the hosts of the testbed back-to-back, disabling the WAN emulator. We then perform two different kinds of measurements to estimate the two main delay components: the forwarding delay and the buffering delay. To estimate the forwarding delay we perform experimentations with *TimeD* configured to use a block depth $= 1$, and we inject UDP packets at all the rates reported in Table 2. With such a block depth, *TimeD* forwards the packets as soon as they enter the queue, and no buffering is performed. We then perform the same experiments without *TimeD* and compare the delays obtained. We observe that the forwarding delay is in the order of 10 μs at all the considered packet rates, we believe this delay can be considered negligible, at least for the operating scenarios we consider.

For the buffering delay, we measure the transfer time experienced by UDP packets using all the rates and interleaving configurations reported in Table 2. The experiments evidence that, thanks to the packet release strategy presented in Section 4.2.3, the delay is typically constant (except few spikes), and in average, it is equal to:

$$\delta_{avg} = (N - 1) \times IPT_{avg} \tag{20}$$

where N is the block size and IPT_{avg} is the average inter-packet time. As already remarked before, such delay has to be carefully taken into account from the application point of view. To provide a real example, if we use *TimeD* with a streaming server – assuming that the stream has a constant packet rate of 720 pps (i.e. frame rate of 24 frames/s, 30 slices per frame, and a single slice per packet) and that packet sizes follow a normal distribution (Garrett and Willinger, 1994) – and with an interleaving block of 48 packets, we obtain an average buffering delay of about 65 ms, which is acceptable in most cases.

5.2.2. *Achievable throughput*

Even if we believe that performance issues have to be carefully taken into account before the deployment of *TimeD* in operational environments, we verify that *TimeD* is able to sustain the rate used for the experiments. In particular, we perform experiments aimed at understanding the maximum throughput that *TimeD* is able to sustain. The results (not reported for space constraints) show that *TimeD* is able to work at full speed (i.e. 100 Mbps) with all the interleaving configurations in Table 2, even when running on the same host as the test application (D-ITG).

5.3. *Loss decorrelation in time varying conditions*

To validate the loss decorrelation capability in time varying conditions, we perform a set of experiments in which we randomly vary the channel conditions imposed by the network emulator using uniform values between 5 → 35 min. During the experiments, we run both *TimeD* and D-ITG: the former interleaves the packets generated by the latter. Fig. 6 shows the results in terms of loss-burst length over time, obtained during a period of about 1.5 h. The channel changes behavior 4 times during the experiment, and *TimeD* accurately estimates the time-varying channel conditions (gray bold line), being always able to set the proper block size. Consequently, the application whose traffic is interleaved experiences a channel with non-bursty losses during the entire experiment (red dashed line). This picture allows to observe the real benefits achievable by a real application using *TimeD*. In the next section we will see what happens in the Big Internet and with different applications.

6. Experimental study

The final step of our analysis is performed in the wild. To experimentally quantify both the amount and the degree of loss burstiness over real networks, we performed a large set of experiments. We considered both wired and wireless networking technologies, as well as local and wide area networks. In this paper, we only show the results obtained over the Internet using PlanetLab (Section 6.2.1) and over a real satellite network (Section 6.3.1). We first verify the improvements achieved with *TimeD* in terms of loss decorrelation (i.e. average loss-burst length reduction). Such benefits can be beneficial to all the applications based on the IP protocol, which are several and heterogeneous, as discussed in Section 1. Then we analyze the benefits achievable by an example real video application, running on top of UDP, in terms of video distortion reduction (Sections 6.2.2 and 6.3.2). Finally, in Section 6.4 we present results on the effectiveness of the packet release strategy described in Section 4.2.3. Before digging into the results obtained, we describe how we evaluated the video distortion (Section 6.1).

6.1. *Video dstortion*

In order to evaluate the distortion on videos caused by both bursty and non-bursty losses, we used the theoretical model proposed in Liang et al. (2003). This model has proved able to estimate the real evolution of the distortion associated to video, in the presence of generic loss patterns, being more accurate than the so called “additive models”. These models do not account for the real loss pattern, as they assume that the total distortion is equal to the sum of the distortions related to isolated losses. In the following we report some of the final results from (Liang et al., 2003) for the estimation of the distortion of the coded video. For the sake of brevity we consider the most simple and representative cases, i.e. single loss, loss burst of length two, and two losses separated by a “lag”. Said $\sigma_s^2[k]$ the mean square error (MSE), i.e. the distortion initially related to the isolated loss of frame k, we can express the total distortion $D_s[k]$ for the case of single

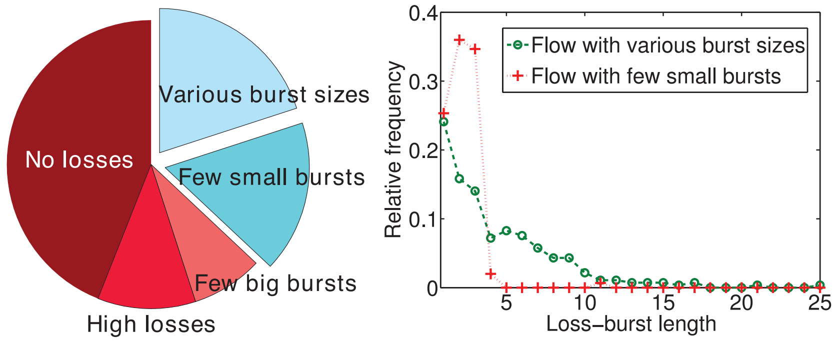


Fig. 7. Loss behaviors (left) and distribution of loss-burst length (right) on PlanetLab.

loss as:

$$D_s[k] = \alpha \cdot \sigma_s^2[k] \qquad (21)$$

where α takes into account the effect of the propagation of the initial error due to intra coding and spatial filtering.

The total distortion in the case of a loss burst of length two is instead equal to:

$$D_s[k-1,k] = \sigma_s^2[k-1] + D_s[k-1] + D_s[k] + 2\rho\sqrt{D_s[k-1]\cdot D_s[k]} \qquad (22)$$

which is a sum of two single distortions, a cross-correlation term, and the initial distortion due to the loss of the frame $k-1$. These last two terms characterize this model as opposed to additive models, and cause its greater effectiveness in representing the distortion.

The last case here reported regards the loss of two frames separated by a number l of frames (“lag”) not greater than N (“intra update period”, that is the distance between two subsequent frames of type I). We note that if $l > N$, the two losses can be considered as independent, and the total distortion as additive. The distortion due to two separated losses localized in $k-l$ and k, with $l \le N$, is given by:

$$D[k-l,k] = \vartheta(l)\cdot D_s[k-l] + \frac{\sigma^2[k]}{\sigma_s^2[k]}D_s[k], \qquad (23)$$

where $\vartheta(l)$ takes into account the error attenuation capabilities of the decoding scheme depending on l, and $\sigma^2[k]$ corresponds to the MSE for frame k due to the contributions from both the loss of frame k and the error propagation related to the loss of frame $k-l$. It is worth noting that the distortion in (23) is a function of the distortions caused by the isolated losses of frames $k-l$ and k. The coefficients of these two distortions, that depend on l and on the correlation between errors, represent another characteristic aspect of this model.

A very strong hypothesis is needed for the application of the model: the initial distortion related to the loss of a specific frame has to be known, being previously measured and stored by simulating the loss event. So there is the need of “*pre-measured distortions*”. In order to obtain such preliminary measurement, we used two widely adopted test video sequences, known as *Foreman* and *Claire*. Each sequence is coded according to the video compression standard JVT/H.2L[8], in format QCIF, and it is 280 frames long, coded at 80 fps with fixed quantization level and 36 dB PNSR[9]. The first frame of each sequence is intra-coded and is followed by P-frames. Every 4 frames a slice is intra-coded in order to reduce the propagation error in case of losses. The intra update period is equal to $N = 4\cdot 9 = 36$ frames. Using these known sequences, we calculate the distortion for the loss of the single frames and use these values for the computation of the total distortion caused by bursts of losses.

Another important hypothesis at the base of our analysis is that each packet contains one encoded video frame, belonging to one of the two reference sequences, whose total distortions related to bursts are known. Finally, another simplifying hypothesis is adopted: we assume that the losses are enough separated to let the distortions related to a generic loss pattern be approximated by the distortions obtained for isolated bursts of losses. This hypothesis is not a limitation because, as shown in Liang et al. (2003), losses separated by more of 10 packets can be considered as isolated, for what concerns the total distortion. Moreover, the distance between bursts is a measured parameter, that can be used to evaluate the validity of such hypothesis. From this assumption, we introduce the metric we used for performance evaluation of the interleaving, the *total normalized distortion* D_{tot}^{norm} related to the transmission of N packets:

$$D_{tot}^{norm} = m_1 + m_2\cdot D_2^{norm} + \ldots + m_N\cdot D_N^{norm}, \qquad (24)$$

where D_i is the distortion due to a loss burst of length i, m_i is the number of occurrences of loss bursts of length i, $D_i^{norm} = D_i/D_1$, $D_1^{norm} = 1$.

A source of uncertainty comes from the method used to obtain the value of coefficients D_i^{norm} with $i = 1,\ldots,N$. In fact, in order to have such values for all i, the missing ones are obtained by *interpolation* or *extrapolation* on measured ones.

6.2. Experiments on the wired internet

In this section we report the experimental evaluation of the performance of *TimeD* on the Internet. For this analysis, we used PlanetLab, the well known, planetary-scale testbed, interconnecting about a thousand nodes spread all over the world. PlanetLab has been selected because it allows us to have a sketch of the performance of the *Big Internet*. We used several sets of nodes, each composed of 50 nodes from the 5 continents and from both industry and academia. This is to average also on the different countries and types of connections. We generated UDP flows with different traffic profiles (CBR and VBR) using D-ITG. The test were repeated 10 times, and the results averaged.

6.2.1. Characteristics of the losses over PlanetLab

Fig. 7 (left) shows the results obtained with CBR UDP flows of 1 Mbps (packet rate of 250 pps and payload size of 512 bytes) and duration of 3 min. The loss behaviors can be grouped into 5 main classes: at the two opposites are 45% of the flows experimenting no losses and 10% of the flows experimenting high losses (> 50% of packets); another 8% of the flows have a low loss rate, with losses concentrated in one or few large bursts (> 100 packets); the remaining 37% of the flows also experience a low loss rate, but their losses are more spread over time and form either bursts of small sizes (17% of flows) or bursts with geometric-like distribution (20% of flows).

Very large loss percentages or rare but large loss bursts are symptoms of problems persistent or temporary on the network, whose solution is out of the scope of this paper. The remaining 37% of the flows

[8] Also known as ITU-T H.264/AVC, corresponding to part 10 of the standard MPEG-4 ISO/IEC 14496–10.

[9] $PSNR = 10\log_{10}(255^2/D_e)$ where D_e is equal to the quantization error.

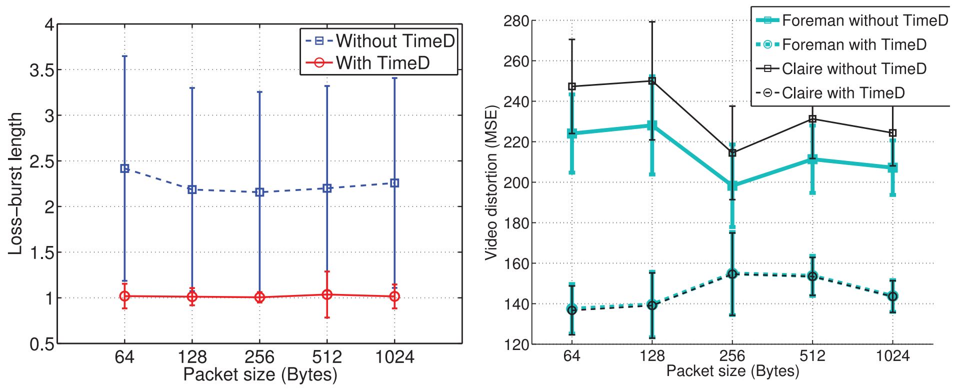


Fig. 8. Results of *TimeD* on PlanetLab.

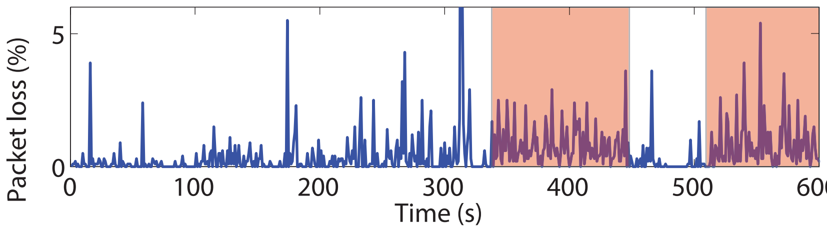


Fig. 9. Time behavior of the losses over the satellite network.

are the ones we target with our approach. Fig. 7 (right) shows the distribution of the loss-burst length for 2 flows from these classes. For one of them, the loss bursts have sizes ranging from 1 to 5 packets. For the other flow, the distribution has a geometric-like shape, with bursts ranging from 1 to about 25 packets. The two flows are examples of the possible distribution of losses over the Internet, indicating that the losses happen in bursts, mostly of small size. Similar considerations can be done with the other traffic profiles. According to these results, our approach can be effectively used in 37% of the cases, and it is even more useful over satellite networks, as shown in Section 6.3.1.

6.2.2. TimeD *results*

We generate UDP flows with different CBR and VBR traffic profiles from a set of selected source/destination pairs. The nodes used for the tests have been selected looking at the results of the analysis reported in the previous section. Fig. 8 shows the results obtained in terms of loss decorrelation (left) and video distortion reduction (right) for one source/destination pair. Unless specifically reported, similar considerations can be made in the other cases. The plot on the left of Fig. 8 shows the average loss-burst size and its standard deviation (the vertical segments) as a function of the packet size (packet rate is 50 pps in this case) when using and not using *TimeD*. As shown, the average loss-burst length reduces from about 2.5 packets per burst to about 1 packet per burst, as in the case of uncorrelated losses. This testified that *TimeD* is able to completely spread the losses. The plot also shows that the standard deviation of the loss bursts is highly decreased when using *TimeD*, which implies higher predictability of the losses.

The right plot of Fig. 8 shows that the video distortion is also highly reduced thanks to *TimeD*, with both video sequences. In particular, in absence of *TimeD*, the normalized distortion varies in 220 → 250 for the Claire video sequence, and in 200 → 230 for the Foreman video sequence. When using *TimeD*, the distortion is always smaller than 160 for both video sequences. Where the losses have the highest correlation (for packet size of 64 bytes) the distortion is reduced about 45% (from 250 to 140). This is a direct consequence of the loss-burst reduction and it means that the benefits can actually be exploited by real applications.

6.3. Experiments on satellite network

Satellite networks are being more and more used for Internet access (for rural areas, in-flight and over-sea connectivity, etc.), while being particularly exposed to bursty losses. For these experiments we use a commercial satellite connection from one of the largest providers in Europe. The satellite connection is bidirectional and has nominal bandwidth of about 3 Mbps in downlink and 300 Kbps in uplink. Our testbed is composed of hosts connected to the Internet through the satellite connection, and servers located in our University (connected to the Internet though the GARR[10]). We generate UDP flows with different CBR and VBR traffic profiles, with average packet sizes ∈ {64; 128; 256; 512; 1024; 1472} bytes and rates ∈ {10; 100; 1000} pps. It is worth noting that with packet size > 512 bytes and rate of 1000 pps the link is completely saturated. Flows have duration of 3 → 10 min and every experiment is repeated 10 times. Before deploying *TimeD* on this infrastructure, we perform a set of measurements to understand the loss characteristics of the satellite channel.

6.3.1. Characteristics of the losses over the satellite link

As anticipated in Section 4.2.1, to choose the measurement period we study the literature and we analyze the time behavior of the losses over the satellite network. In Tao et al. (2004) the authors perform a long-term measurement campaign, and report the long-term delay and loss rate measured over different paths connecting the same hosts. The results show that there is high variability in the loss rate, and, if observed with a resolution of 1 min (i.e. computing the average loss rate over 1 min intervals), the paths look quite different. They also show that the average value of the loss-variation period is 1 → 4 min. Fig. 9 shows a zoom of the samples of the packet loss over the satellite network, during a period of 10 min, sending packets of 128 bytes at 1000 pps. As we can see, there are periods (e.g. between 350 and 450 s, and between 520 and 600 s) in which the loss rate is larger than that in the other periods (e.g. between 0 and 80 s). We also observe that the duration of such *good* and *bad* periods is generally between 50 and 100 s. Tests performed with other traffic profiles

[10] http://www.garr.it

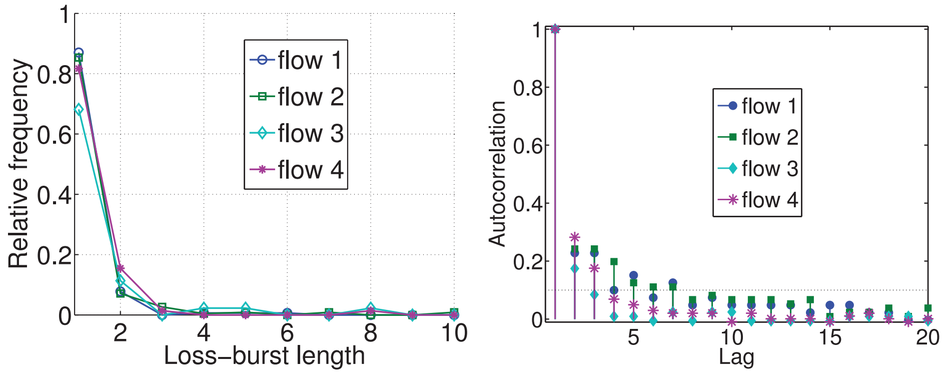


Fig. 10. Loss burst distribution and autocorrelation over the satellite network.

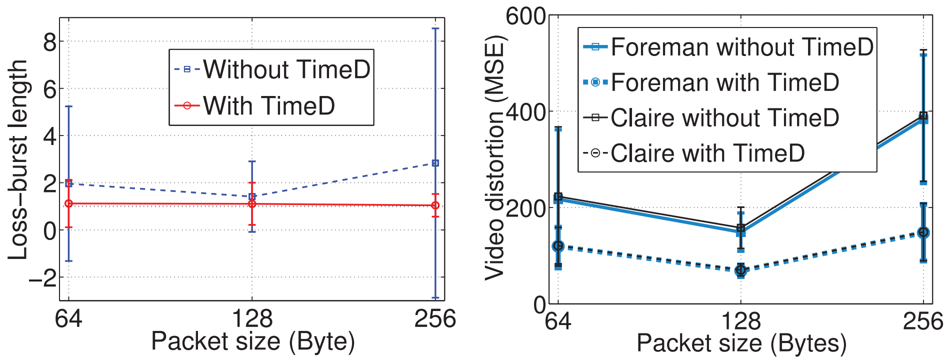


Fig. 11. Results of *TimeD* on satellite network.

evidence the same behavior. This result, confirming previous results in literature (Tao et al., 2004), motivates our choice of evaluating the status of the network every 1 min.

Regarding the burstiness of the losses, 50% of the flows experience losses over this network scenario, and they can be divided in 4 classes. Fig. 10 shows the typical loss-burst length distribution (left) and loss autocorrelation (right) of flows – in the downlink direction – belonging to the 4 classes. To study the autocorrelation of the loss process, we follow the methodology proposed in Yajnik et al. (1999).

As shown in the left plot of Fig. 10, for all the flows the loss burst lengths range from 1 to 5 packets. On the other hand, as shown in the right plot of Fig. 10, the loss process presents a high autocorrelation, which means that the losses are indeed bursty. For example, fitting the loss process of such flows with a 2-MC[11] gives π_b of 0.002 $\rightarrow$ 0.005 and ρ of 30 $\rightarrow$ 400. Such results indicate that the losses over the satellite link are not frequent but highly correlated. This means that in this scenario *TimeD* can provide high benefits, as shown in the next section.

6.3.2. TimeD *results*

We experimentally verify the improvements achieved with *TimeD* in terms of average loss-burst length reduction. The left plot of Fig. 11 is related to CBR traffic having packet size $\in$ {64, 128, 256} bytes and packet rate of 1000 pps. We observe that, when not using *TimeD*, the average loss-burst length on the link is between 1.5 and 3 packets, with a very high standard deviation (see the vertical segments). The use of *TimeD* allows to reduce both the average loss-burst length and the related standard deviation. As a result, the loss-burst length is always smaller than 1.12, and the loss correlation at lag 2 (not shown here for space constraints) is always smaller than 0.01. The experiments performed with the other traffic profiles evidence similar results. This means that, thanks to its features, *TimeD* is able to decorrelate the losses in real networks, allowing the applications to experiment a channel that is close to the Bernoulli one.

Finally, we analyze the benefits achievable by a real application using the video distortion model presented in Section 6.1. The right plot of Fig. 11 shows the video distortion estimated by the model for two test video sequences subject to the losses reported in the left plot of the same figure (with and without *TimeD*). As we can see, in accordance with the model, the distortion increases with the loss-burst length. Consequently, the video flows whose packets are interleaved by *TimeD* experience smaller average and standard deviation values of the video distortion. Where the losses are more correlated (for packet size of 256 bytes) *TimeD* improves the distortion of more than 65% (from 450 to 150).

Summarizing, we can say that *TimeD* highly improves the performance of applications such as those for video communications, providing higher Quality of Experience.

6.4. *Packet release strategy*

Before closing the paper, we present results on the effectiveness of the packet release strategy described in Section 4.2.3. Fig. 12 shows the time behavior of the bitrate of VBR traffic flows with exponentially distributed packet rate (average value is 100 pps) and constant packet size of 512 bytes, over Satellite network. The different lines in the figure show the profiles of the injected traffic and of the received traffic when *TimeD* enables and disables the packet release strategy. The continuous spikes in the bitrate of the flow received when such strategy is disabled are due to the sudden buffer emptying. On the contrary, when the packet release strategy is enabled, *TimeD* closely follows the original traffic profile.

[11] As the correlation persists also at lags > 2, fitting the process with a Markov Chain with a higher number of states is also possible. However, 2-MC represents the best compromise between accuracy and complexity.

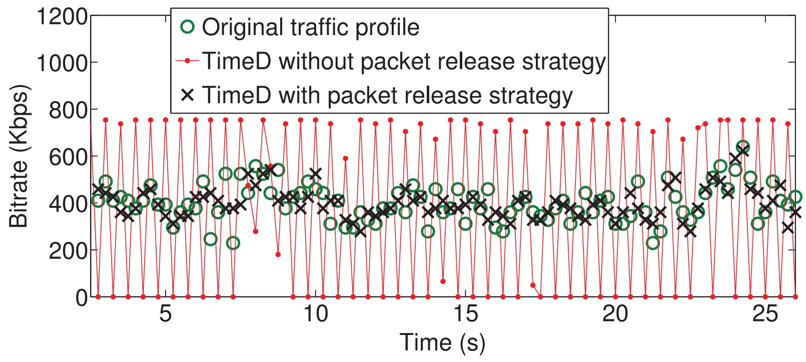


Fig. 12. Benefit of the packet release strategy.

7. Conclusion

Most of the related works either proposing the use of interleaving, or other techniques to cope with bursty losses, adopted theoretical or simulative approaches. When proposing a real implementation and experimentation of the interleaving, they were not meant to work with other applications than the one targeted. On the contrary, our a solution is able to work with all the possible applications based on the Internet Protocol, providing benefits to the (large) body of them that suffer from bursty losses. To reach this target, in this work we studied through a comprehensive analysis the possibility to deploy time diversity at IP level in real networks. Firstly, we analytically derived how to properly configure a block interleaving as a function of the network losses. Secondly, we confirmed the analytical results through a simulator purposely developed, evidencing also possible counter effects. Then, we presented our tool, called *TimeD*, for deploying time diversity at the IP level, analyzing, discussing, proposing and integrating in *TimeD* effective solutions for several issues arising from its use in real environments. We validated the tool in emulation, underlining how it allows to achieve the expected results thanks to its particular features. Afterwards, we moved on real wired and wireless networks, where we studied the burstiness of losses, providing evidences and quantitative evaluations. Finally, we used *TimeD* on such real networks, demonstrating its positive impact on loss burstiness in general and on the performance of a UDP-based video application.

In this paper, we analyzed the benefits achievable with *TimeD* explicitly targeting UDP traffic. Moreover, we performed experiments using, as a test case, a video application. We remark, however, that *TimeD* has been conceived to be used at the IP level, independently on the upper- and lower-layer protocols. While the buffering delay introduced by the interleaver naturally suits the requirements of non-interactive applications (e.g. audio and video streaming), we believe that other classes of applications – like for example DNS or uTP-based applications – can possibly profit from the use of *TimeD*.

Our ongoing work is concerned with the use of *TimeD* with other applications and network technologies. Moreover, we are studying the interactions between *TimeD* and transport protocols adopting congestion control strategies (SCTP, DCCP, and TCP) and we are performing a careful analysis of the interleaving applied on bidirectional traffic using *TimeD* on both ends of network paths.

Finally, as for the use of *TimeD* in specific application scenarios, we have two ongoing activities: (i) we are experimenting it into home routers of the BISmark (Sundaresan et al., 2011) project and (ii) we plan to release it as a userspace library that applications running on a host can use.

Acknowledgments

This work is partially funded by the MIUR projects: PLATINO (PON01 01007), SMART HEALTH (PON04a2 C), S2-MOVE (PON04a3 00058), SIRIO (PON01 02425), and art. 11 DM 593/2000 for NM2 srl.

References

http://traffic.comics.unina.it, last accessed 12 Aug 2015.
Apostolopoulos, J., Trott, M., 2004. Path diversity for enhanced media streaming. Commun. Mag., IEEE 42 (8), 80–87. doi:10.1109/MCOM.2004.1321395.
Bolot, J.-C., Crépin, H., Vega-García, A., 1995. Analysis of audio packet loss in the internet. In: NOSSDAV.
Botta, A., 2009. Towards Informed Diversity in IP Networks: Techniques, Issues, and Solutions. University of Napoli Federico II Ph.D. thesis. http://wpage.unina.it/a.botta, last accessed Aug 12, 2015.
Botta, A., Dainotti, A., Pescapé, A., 2012. A tool for the generation of realistic network workload for emerging networking scenarios. Comput. Netw. 56 (15), 3531–3547.
Bui, V., Zhu, W., Botta, A., Pescapé, A., 2010. A Markovian approach to multipath data transfer in overlay networks. IEEE Trans. Parallel Distrib. Syst., 21 (10), 1398–1411. doi:10.1109/TPDS.2010.13.
Cai, J., Chen, C.W., 2001. Fec-based video streaming over packet loss networks with pre-interleaving. In: Proceedings of the International Conference on Information Technology: Coding and Computing.
Casu, F., Cabrera, J., Jaureguizar, F., Garcia, N., 2012. Low latency ldgm code for multimedia-packet stream in bursty packet loss networks. In: Proceeedings of the IEEE International Conference on Consumer Electronics (ICCE), 2012, pp. 63–64. doi:10.1109/ICCE.2012.6161738.
Chen, L.-J., Sun, T., Sanadidi, M., Gerla, M., 2004. Improving wireless link throughput via interleaved FEC. In: IEEE ISCC.
Chin, S.K., Braun, R., 2001. Improving video quality using packet interleaving, randomisation and redundancy. In: IEEE LCN.
Chung, S., Agrawal, D., Kim, M., Hong, J., Park, K., 2004. Analysis of bursty packet loss characteristics on underutilized links using SNMP. IEEE/IFIP E2EMON.
Claypool, M., Zhu, Y., 2003. Using interleaving to ameliorate the effects of packet loss in a video stream. In: IEEE ICDCSW.
Cui, H., Qian, D., Zhang, X., Wu, Y., Wang, L., 2012. Network coding-based rate allocation and bursty loss protection for video streaming over wireless multi-hop networks. In: Proceedings of the IEEE 12th International Conference on Computer and Information Technology, 2012. IEEE Computer Society, Washington, DC, USA, pp. 684–689. doi:10.1109/CIT.2012.143.
Dang, T.D., Perényi, M., Gefferth, A., Molnár, S., 2006. On the identification and analysis of p2p traffic aggregation. In: Networking.
De Alwis, C., Kodikara Arachchi, H., De Silva, V., Fernando, A., Kondoz, A., 2012. Robust video communication using random linear network coding with pre-coding and interleaving. In: Proceedings of the 19th IEEE International Conference on Image Processing (ICIP), 2012, pp. 2269–2272. doi:10.1109/ICIP.2012.6467348.
Elliott, E., 1963. Estimates of error rates for codes on burst-noise channels. Bell Syst. Tech. J 42 (9), 1977–1997.
Ellis, M., Pezaros, D., Kypraios, T., Perkins, C., 2012. Modelling packet loss in rtp-based streaming video for residential users. In: Proceedings of IEEE 37th Conference on Local Computer Networks (LCN), 2012, pp. 220–223. doi:10.1109/LCN.2012.6423613.
Ellis, M., Pezaros, D.P., Kypraios, T., Perkins, C., 2014. A two-level Markov model for packet loss in udp/ip-based real-time video applications targeting residential users. Comput. Netw. 70 (0), 384–399.
Finamore, A., Mellia, M., Meo, M., Rossi, D., 2010. Kiss: stochastic packet inspection classifier for udp traffic, 18, pp. 1505–1515.ISSN: 1063-6692.
Garrett, M.W., Willinger, W., 1994. Analysis, modeling and generation of self-similar vbr video traffic. ACM SIGCOMM Comput. Commun. Rev. 24 (4), 269–280.
Ghita, D., Nguyen, H.X., Kurant, M., Argyraki, K.J., Thiran, P., 2010. Netscope: practical network loss tomography. In: INFOCOM. IEEE, pp. 1262–1270.
Gilbert, E., et al., 1960. Capacity of a burst-noise channel. Bell Syst. Tech. J 39 (9), 1253–1265.
Han, D., Anand, A., Akella, A., Seshan, S., 2012. Rpt: re-architecting loss protection for content-aware networks. In: NSDI, pp. 71–84.

Hasegawa, Y., Yamaguchi, I., Hama, T., Shimonishi, H., Murase, T., 2005. Improved data distribution for multipath TCP communication. In: Proceedings of IEEE Global Telecommunications Conference, 2005. GLOBECOM '05., 1, p. 5. doi:10.1109/GLOCOM.2005.1577632.
He, J., Rexford, J., 2008. Toward internet-wide multipath routing. Network, IEEE 22 (2), 16–21. doi:10.1109/MNET.2008.4476066.
Jelassi, S., Rubino, G., 2011a. A comparison study of automatic speech quality assessors sensitive to packet loss burstiness. In: Proceedings of Consumer Communications and Networking Conference (CCNC), 2011 IEEE, pp. 415–420. doi:10.1109/CCNC.2011.5766503.
Jelassi, S., Rubino, G., 2011b. A study of artificial speech quality assessors of voip calls subject to limited bursty packet losses. EURASIP J. Image Video Process. 2011 (1). doi:10.1186/1687-5281-2011-9.
Jelassi, S., Rubino, G., 2013. A perceptually sensitive Markovian model of packet loss processes during VOIP conversations. In: Proceedings of the 9th International Conference on Wireless Communications and Mobile Computing (IWCMC), 2013, pp. 964–969. doi:10.1109/IWCMC.2013.6583687.
Lee, C., Lee, D., Moon, S., 2012. Unmasking the growing UDP traffic in a Campus Network. In: PAM 2012.
Lee, J.Y., Radha, H., 2005. Interleaved source coding (ISC) for predictive video coded frames over the Internet. In: IEEE ICC.
Lee, J.Y., Radha, H., 2006. Evaluation of interleaved source coding (ISC) over channel with memory. In: Proceedings of Conference on Information Sciences and Systems.
Liang, Y., Apostolopoulos, J., Girod, B., 2003. Analysis of packet loss for compressed video: does burst-length matter? In: IEEE ICASSP.
Liang, Y.J., Apostolopoulos, J.G., Girod, B., 2002. Model-Based Delay-Distortion Optimization. In: Proceedings of Asilomar Conference on Signals, Systems, and Computers.
Liu, F., Luan, T.H., Shen, X.S., Lin, C., 2014. Dimensioning the packet loss burstiness over wireless channels: a novel metric, its analysis and application. Wirel. Commun. Mob. Comput. 14 (12), 1160–1175. doi:10.1002/wcm.2262.
Liu, T., Wang, Y., Boyce, J., Yang, H., Wu, Z., 2009. A novel video quality metric for low bit-rate video considering both coding and packet-loss artifacts. IEEE J. Sel. Top. Signal Process. 3 (2), 280–293.
Mehmood, M.A., Sarrar, N., Uhlig, S., Feldmann, A., 2013. Impact of access bandwidth on packet loss: a flow-level analysis. In: Proceedings of the IEEE Malaysia International Conference on Communications (MICC), 2013. IEEE, pp. 259–264.
Nagano, T., Ito, A., 2014. Packet loss concealment of voice-over IP packet using redundant parameter transmission under severe loss conditions. J. Inf. Hiding and Multimed. Signal Process. 5 (2), 285V294.
Nguyen, H., Roughan, M., 2013. Rigorous statistical analysis of internet loss measurements. IEEE/ACM Trans. Netw. 21 (3), 734–745. doi:10.1109/TNET.2012.2207915.
Nguyen, H.X., Roughan, M., 2010. Rigorous statistical analysis of internet loss measurements. In: SIGMETRICS, pp. 361–362.
Nguyen, T., Mehra, P., Zakhor, A., 2003. Path diversity and bandwidth allocation for multimedia streaming. In: Proceedings of the 2003 International Conference on Multimedia and Expo ICME '03. IEEE Computer Society, Washington, DC, USA, pp. 1–4.
Oztas, B., Pourazad, M., Nasiopoulos, P., Leung, V., 2012. A study on the hevc performance over lossy networks. In: Proceedings of 19th IEEE International Conference on Electronics, Circuits and Systems (ICECS), 2012, pp. 785–788. doi:10.1109/ICECS.2012.6463542.
Paxson, V., 1997. End-to-end Internet packet dynamics. SIGCOMM Comput. Commun. Rev. 27 (4), 139–152. http://doi.acm.org/10.1145/263109.263155.
Paxson, V., 1999. End-to-end internet packet dynamics. IEEE/ACM Trans. Netw. 7 (3), 277–292.
Ribeiro, B., e Silva, E., Towsley, D., 2005. On the Efficiency of Path Diversity for Continuous Media Applications. UMass CMPSCI Technical Report 05-19.
Rodney, C., 1974. Stochastic Processes. Allen & Unwin Ltd, London.
Salvo Rossi, P., Palmieri, F., Iannello, G., Petropulu, A., 2004. Packet interleaving over lossy channels. In: Proceedings of IEEE International Symposium on Signal Processing and Information Technology.
Soloducha, M., Raake, A., 2014. Speech quality of VOIP: bursty packet loss revisited. In: Proceedings of ITG Symposium on Speech Communication; 11, pp. 1–4.
Sundaresan, S., de Donato, W., Feamster, N., Teixeira, R., Crawford, S., Pescapè, A., 2012. Measuring home broadband performance. Commun. ACM 55 (11), 100–109.
Tao, S., Guérin, R., 2004. Application-specific path switching: a case study for streaming video. In: Proceedings of the 12th Annual ACM international conference on Multimedia MULTIMEDIA '04:. ACM, New York, NY, USA, pp. 136–143. http://doi.acm.org/10.1145/1027527.1027553.
Tao, S., Xu, K., Xu, Y., Fei, T., Gao, L., Guerin, R., Kurose, J., Towsley, D., Zhang, Z.-L., 2004. Exploring the performance benefits of end-to-end path switching. SIGMETRICS Perform. Eval. Rev. 32 (1), 418–419. http://doi.acm.org/10.1145/1012888.1005746.
Vergetis, E., Guérin, R., Sarkar, S., 2005. Improving performance through channel diversity in the presence of bursty losses. In: ITC-19.
Vergetis, E., Pierce, E., Blanco, M., Guérin, R., 2006. Packet-level diversity - from theory to practice: an 802.11-based experimental investigation. In: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking MobiCom '06:. ACM, New York, NY, USA, pp. 62–73. http://doi.acm.org/10.1145/1161089.1161098.
Wah, B., Lin, D., 1999. Transformation-based reconstruction for real-time voice transmissions over the internet. IEEE Trans. Multimed. 1 (4), 342–351.
Wang, L., Gelenbe, E., 2014. An implementation of voice over IP in the cognitive packet network. In: Information Sciences and Systems 2014. Springer, pp. 33–40.
Wang, W., Wu, C., Ohzahata, S., Kato, T., 2014. Experimental analysis of TCP behaviors against bursty packet losses caused by transmission interruption. ICN 2014 147.
Wang, Y.A., Huang, C., Li, J., Ross, K.W., 2009. Queen: estimating packet loss rate between arbitrary internet hosts. In: PAM '09. Springer-Verlag, Berlin, Heidelberg, pp. 57–66.
Wei, D.X., Cao, P., Low, S.H., 2007. Packet loss burstiness: measurements and implications for distributed applications. IEEE IPDPS.
Yajnik, M., Kurose, J., Towsley, D., 1996. Packet loss correlation in the MBone multicast network. In: IEEE GLOBECOM.
Yajnik, M., Moon, S., Kurose, J., Towsley, D., 1999. Measurement and modelling of the temporal dependence in packet loss. In: IEEE INFOCOM.
Yao, J.-H., Chen, Y.-M., 1997. Experiments of real-time MPEG audio over the internet. Fujitsu Sci. Tech. J. 33 (2), 138–144.
Zhang, M., Dusi, M., John, W., Chen, C., 2009. Analysis of UDP traffic usage on internet backbone links. In: SAINT.
Zhang, X., Xu, Y., Hu, H., Liu, Y., Guo, Z., Wang, Y., 2012. Profiling skype video calls: rate control and video quality. In: Proceedings of IEEE INFOCOM, 2012. IEEE, pp. 621–629.

Alessio Botta. Alessio Botta is a PostDoc at the Department of Computer Engineering and Systems of the University of Napoli Federico II (Italy). He graduated in Telecommunications Engineering (M.S.) and obtained the Ph.D. in Computer Engineering and Systems, both at University of Napoli Federico II. His research interests are in the area of networking and, in particular, in the area of network performance measurement and improvement, with a specific focus on wireless and heterogeneous systems. Alessio Botta has coauthored more than 40 international journal (IEEE Communications Magazine, IEEE Transactions on Parallel and Distributed Systems, Elsevier Computer Networks, etc.) and conference (IEEE Globecom, IEEE ICC, IEEE ISCC, etc.) publications. He has served and serves several technical program committees of several international conferences (IEEE Globecom, IEEE ICC, etc.) and he acts as reviewer for different international conferences (IEEE Infocom, etc.) and journals (IEEE Transactions on Mobile Computing, IEEE Network, IEEE Transactions on Vehicular Technology, etc.) in the area of networking. In 2010 he was awarded with the best local paper award at IEEE ISCC 2010. Alessio Botta has served and serves as independent reviewer of research and implementation project proposals for the Romanian government.

Antonio Pescapé. Antonio Pescapé [SM '09] received the M.S. Laurea Degree in Computer Engineering and the Ph.D. in Computer Engineering and Systems at University of Napoli Federico II, Napoli (Italy). He is currently an Associate Professor at the Department of Electrical Engineering and Information Technology of the University of Napoli Federico II (Italy) where he teaches courses in Computer Networks, Computer Architectures, Programming, and Multimedia and he has also supervised and graduated more than 160 among BS, MS, and PhD students. His research interests are in the networking field with focus on Internet Monitoring, Measurements and Management and on Network Security. Antonio Pescapé has co-authored over 140 journal (IEEE ACM Transaction on Networking, Communications of the ACM, IEEE Communications Magazine, JSAC, IEEE Wireless Communications Magazine, IEEE Networks, etc.) and conference (SIGCOMM, Infocom, Conext, IMC, PAM, Globecom, ICC, etc.) publications and he is co-author of a patent. He has served and serves as workshops and conferences Chair (including IEEE ICC (NGN symposium)) and on more than 100 technical program committees of IEEE and ACM conferences. He serves as Editorial Board Member of Journal of Network and Computer Applications and has served as Editorial Board Member of IEEE Survey and Tutorials (2008–2011) and was guest editor for the special issue of Computer Networks on "Traffic classification and its applications to modern networks" and for the special issue of Journal of Future Generation Computer Systems on "Internet of Things and Cloud Services". For his research activities he has received several awards, comprising a Google Faculty Award, several best paper awards and two IRTF (Internet Research Task Force) ANRP (Applied Networking Research Prize). Antonio Pescapé has served and serves as independent reviewer/evaluator of research and implementation projects and project proposals co-funded by the EU Commission, Sweden government, several Italian local governments, Italian Ministry for University and Research (MIUR) and Italian Ministry of Economic Development (MISE). Antonio Pescapé is a Senior Member of the IEEE.