A Markov Framework for Error Control Techniques Based on Selective Retransmission in Video Transmission over Wireless Channels

Leonardo Badia, *Member, IEEE,* Nicola Baldo, *Member, IEEE,* Marco Levorato, *Member, IEEE,* Michele Zorzi, *Fellow, IEEE*

***Abstract*—We present a framework, based on Markov models, for the analysis of error control techniques in video transmission over wireless channels. We focus on retransmission-based techniques, which require a feedback channel but also enable to perform adaptive error control. Traditional studies of these methodologies usually consider a uniform stream of data packets. Instead, video transmission poses the non-trivial challenge that the packets have different sizes, and, even more importantly, are incrementally encoded; thus, a carefully tailored model is required. We therefore proceed on two different sides. First, we consider a low-level description of the system, where two main inputs are combined, namely, a video packet generation process and a wireless channel model, both described by Markov Chains with a tunable number of states. Secondly, from a high-level perspective, we represent the whole system evolution with another Markov Chain describing the error control process, which can feed the packet generation process back with retransmissions. The framework is able to evaluate hybrid automatic repeat request with selective retransmission, but can also be adapted to study pure automatic repeat request or forward error correction schemes. In this way, we are able to comparatively evaluate different solutions for video transmission, as well as to quantitatively assess their performance trends in a variety of different scenarios. Thus, our framework can be used as an effective tool to understand the behavior of error control techniques applied to video transmission over wireless, and eventually identify design guidelines for such systems.**

***Index Terms*—Hybrid automatic repeat request, video transmission, channel coding, Markov analysis.**

I. Introduction

The growing interest in video transmission over the wireless channel, which is an inherently unreliable and lossy medium, poses several challenges for the design of error control techniques. In this paper, we discuss Hybrid Automatic Repeat reQuest (hybrid ARQ) [1], which is a combination of Forward Error Correction (FEC) and Automatic Repeat reQuest (ARQ, hereafter called “plain ARQ” to distinguish it from the hybrid version). Pure FEC may introduce excessive redundancy and reduce the available bandwidth, whereas ARQ, which requires a return control channel, may lead to increased delay. Hybrid ARQ tries to avoid these problems with a combined approach: data are protected by error-correcting codes that repair some of the errors introduced by the channel (FEC approach), and those data parts which are still in error even after FEC decoding are retransmitted upon request (ARQ approach) [2]. Many authors suggest that hybrid ARQ transmission schemes outperform FEC and plain ARQ for multimedia [3], [4]. However, hybrid ARQ still requires feedback exchanges and retransmissions; thus, it needs to be carefully designed to avoid excessive delays.

The present paper introduces an analytical methodology based on Markov chains to study error control techniques for video content delivery over wireless. Even though the main focus is on hybrid ARQ, *all* previously mentioned error control techniques can be framed in the analysis. Moreover, we propose a specific hybrid ARQ scheme tailored on video contents, and evaluate its performance.

A. Video transmission through incremental encoding

Video flows could be transmitted as sequences of independent still frames, as done when adopting the Joint Photographic Experts Group (JPEG) standard [5] in a manner that is informally referred to as M-JPEG (Multiple JPEG), i.e., just transmitting subsequent JPEG pictures. This approach is limited to very static flows; more refined techniques, such as the Moving Pictures Expert Group (MPEG) standards, use inter-frame prediction to exploit correlation among frames [6]. MPEG standards use differential encoding mechanisms decomposing the video flow into *frames* of different kinds. The usual classification comprises intracoded (I), forward predicted (P) or bidirectionally predicted (B) frames. An I frame contains a self-standing static picture, where texture values are coded using the Discrete Cosine Transform (DCT). P and B frames are predicted from the closest match with the preceding I (or P) frame, or both the preceding and the subsequent I (or P) frames, respectively, using motion vectors. Prediction errors are transformed with DCT, and the resulting coefficients are quantized and coded with variable length. I, P, and B frames are arranged in a periodic pattern referred to as a group of pictures (GoP). Note that a GoP entirely consisting of I frames will turn the MPEG into a variant of the aforementioned M-JPEG.

For the sake of analytical tractability, we assume that the video flow consists of units called *packets*, divided in two categories, which we call *independent* and *dependent* packets. The term “packet” must be regarded here as an encoding

L. Badia is with IMT Lucca Institute for Advanced Studies, piazza S. Ponziano 6, 55100 Lucca, Italy. N. Baldo is with CTTC, Parc Mediterrani de la Tecnologia, Av. Canal Olímpic, 08860 Castelldefels, Barcelona, Spain. M. Levorato, and M. Zorzi are with Dept. of Information Engineering, University of Padova, via Gradenigo 6/B, 35131 Padova, Italy. e-mail: {badia,baldo,levorato,zorzi}@dei.unipd.it.

data unit; further packet fragmentation may be introduced by lower layers, as will be argued in II-A. A data unit of the former kind, called in short an *I packet*, represents some self-standing information content (the choice of the name is intentional, as it somehow relates to an I frame of the MPEG standard). Conversely, a dependent packet, or *D packet*, supplies additional details to one I packet of reference, from which it is incrementally encoded. Thus, it requires the correct knowledge of this I packet to be decoded. As a result, we take the video source as alternating between I and D packets. This can be thought of in analogy with MPEG as if every I frame were mapped into a single I packet and all subsequent P and B frames had been joined in a single D packet. This simplification neglects that the last B frame of a GoP depends on two I packets; this could be considered at the price of writing more cumbersome equations. Moreover, this approach is used in related analytical contributions [7], and is similar to the network abstraction layer of an H.264 codec [8].

B. Contributions of the present paper

As a first contribution, we discuss the role of packet differentiation and its implications on the retransmission scheme. In particular, it seems sensible to think of an I packet as worth of retransmission in case of errors, as its reception is necessary to be able to decode the frames, belonging to a D packet, that are incrementally encoded from it. In turn, D packets comprise multiple frames relative to the same I packet; if we are able to tell the erroneous frames apart, we can discard them and keep only the correct ones.

This leads us to consider a hybrid ARQ scheme tailored on video transmission, where both I and D packets can be protected by FEC, but only I packets are selectively retransmitted when in error. A small maximum number of transmission attempts is set in order to guarantee a bounded delay. For erroneous D packets, we assume instead to isolate and discard the incorrect frames, keeping only the good ones. We remark that there exist unequal error protection schemes where FEC codes of different strengths are applied to the packets; our proposal extends this rationale to the ARQ part as well. Remarkably, such a scheme is simple to implement, but exhibits good performance when applied to video transmission over wireless. In particular, our results contradict the common belief that ARQ schemes are not applicable to video due to their excessive delay. Indeed, our proposed selective retransmission mechanism is shown to be often more effective than a stronger error correction code, since it requires to protect only I packets, which amount to a small fraction of the whole data.

Another goal of this paper is to present an analytical and modular evaluation of error control techniques for video over wireless. To this end, we consider a Markov-based approach describing the joint evolution of the video source, the wireless channel, and the retransmission mechanism. This last part of the transmission scheme can be represented via a finite-state machine whose transitions follow a proper Discrete-Time Markov Chain (DTMC), which we refer to as the *macroscopic description* of the ARQ scheme. For the other components of the system, we use Markov models, which have been extensively adopted in the literature to represent the video flow [9] and the wireless channel [10]. Importantly, the source and channel models should take into account correlation among video packets or among the wireless channel errors, something that, if properly tuned, Markov models do with good accuracy. With minor modifications the model can be used for several transmission schemes, i.e., both Type I and Type II ARQ [2], including, as special cases, also pure FEC and plain ARQ.

We believe that our model can serve as a practical yet entirely analytical tool to identify guidelines for related standards and protocols. Thanks to its modularity, different packet structures and error control schemes can be quantitatively compared for any scenario of interest. Furthermore, the model can be also employed in the context of cross-layer optimization for video over wireless. From this perspective, it is possible to extend the present model to also include rate optimization and adaptive modulation and coding techniques of next generation wireless systems, which will exploit the knowledge of the system state to further enhance video transmission.

C. Related work

The present paper has points of contact with the existing literature on error control, especially for what concerns those papers addressing video systems and/or proposing analytical models. The general framework adopted in this paper considers a data stream sent over a wireless channel represented through a Markov chain. Retransmissions are accounted for by considering a larger chain representing the whole system, and several performance metrics are computed by elaborating on the steady-state probability of the system chain. In this sense, the paper is reminiscent of [11]–[13] which use similar methodologies. There are even extensions to hybrid ARQ techniques such as [14], where the "error level" model discussed in II-B is introduced. However, all these papers do not focus on video, and therefore consider a uniform stream of undifferentiated data. Our contribution is a significant step forward since we consider multiple kinds of data units which have different roles for the delivery of multimedia content.

An influential work in this context is represented by [15], which explicitly aims at analyzing HARQ for video flows. Among its contributions, there is an explicit extension to differentiated video packets. However, this case is only studied by means of simulation, whereas the analysis refers to packets with homogeneous profile. Our contribution is different since we present entirely analytical results for this case; moreover, we also use a more advanced model for error correcting codes of different strengths.

Another source of inspiration for the present paper has been [7] (and other work by the same group of authors, e.g., [16]), where an analytical evaluation of error control for video content over wireless channels, again modeled through Markov chains, is performed. However, only FEC is investigated; our paper can be seen as an extension to considering also data retransmission. Although in [7] incremental encoding is taken into account, there is no explicit differentiation between data units; we extend this point by considering I and D packets with actual different roles. Another similarity with our approach is that an analytical evaluation of video signal distortion is presented. Our model used in IV-C is a direct extension of this, and our assumptions, e.g., about Gaussian pixel errors or codec linearization come from these authors. Yet, our contribution is novel since we significantly extend this model by considering the subdivision of the data into two different kinds of packets. Also, a relevant background reference for the video generation model has been [9]. This paper proposes a Markov model for video flows, which we embed into our Markov framework, and is the source for our choices of the distribution of the packet sizes and the correlation values.

Other stimulating impulses to our work were given by [3], where unequal error protection based on non uniform ARQ-FEC is proposed. Although this paper is on mobile multicast in general, and not specifically on video, the idea of applying hybrid ARQ in a non-uniform manner is somewhat similar to ours. Our approach is different since we consider data flows with differentiated packets and explicitly take this aspect into account in the design of hybrid ARQ. Finally, another important related paper is [4], which investigates the optimization of the application layer error-control by jointly designing source coding and FEC. Differently from this paper, we explicitly consider differentiated packets and correlated errors. Note that we do not target an optimization framework for the wireless physical layer; our proposed hybrid ARQ solution is orthogonal to any dynamic adaptation to channel and source conditions. More in general, our study can serve as a basis for further extensions based on similar reasonings.

D. Outline of the present paper

The rest of this paper is organized as follows. Section II describes the transmission system. Section III contains the model for Type II hybrid ARQ, which can be utilized also for other error control schemes. In Section IV we solve the resulting Markov chain and derive some performance indicators. Section V reports numerical evaluations, comparing several schemes, also with reference to a real video trace simulation. Finally, we draw the conclusions in Section VI.

II. Transmission System Description

We consider the transmission of video content over a wireless channel with a feedback link, which may even consist of a control channel with limited bandwidth, for acknowledgement exchange. The video content is subdivided into packets belonging to either of two categories, namely, I packets or D packets. An I packet represents a self-standing frame, whereas a D packet contains a group of incrementally encoded frames. For simplicity, we consider that, after every I packet, only a single D packet is generated, and it refers to this one I packet only. The video source located at the sender's side produces a saturated flow with alternation between I and D packets; packets are denoted as I(1), D(1), I(2), D(2), and so on.

The feedback about the packet transmission outcome is assumed to be error-free, as commonly done in the literature [4], [12]–[14]. The analysis can be extended to the case of erroneous feedback by following the approach reported in [17]; basically, the error-rate of the feedback channel reflects into a direct increase of the packet error probability, without significant quantitative or qualitative changes. For a detailed discussion on imperfect feedback, also considering the case where it is used to adapt modulation and coding parameters, see also [18]. Yet, we explicitly account for the fact that the feedback is non-instantaneous.

TABLE I
List of Symbols Used in the Paper

Video packet generation parameters	
Λ_0, Λ_1, Λ_m	minimum, maximum, average size of an I packet
Δ_0, Δ_1, Δ_m	minimum, maximum, average size of a D packet
$\lambda_{\mathrm{I}}(t\|s)$, $\lambda_{\mathrm{D}}(t\|s)$	distribution of length t of I (resp., D) packets conditioned to the length s of the last I packet
$\mathbf{Q} = (q_{ij})$	steady-state I packet length distribution matrix
$\widetilde{\mathbf{Q}} = (\widetilde{q}_{ij})$	one-step I packet length distribution matrix
ρ	correlation of I packet length
F	number of frames in a GoP (I frame period)
Parameters of Markov channel	
$\mathbf{P} = (p_{ij})$	channel transition matrix
$c, \mathcal{X}$	channel state, set of channel states
ε	average error probability of a slot
B	average length (in slots) of a burst of errors
$L+1$	number of error levels (from 0 to L)
θ_{I}, θ_{D}	thresholds for error correction, for I and D packets
$\psi(k, n\|j, t)$	probability of n errors and end state k conditioned to starting state j and length t
$\varphi(k, \ell\|j, t)$	probability of error level ℓ and end state k conditioned to starting state j and length t
Parameters of the system DTMC	
ς	ARQ stage
s_1, ℓ_1	size and error level of the packet of interest
s_2, ℓ_2	size and error level of the pending packet
Performance evaluation metrics	
τ	average transition time of the system Markov chain
Ω	delay to transmit a GoP
Θ, Υ	throughput, goodput
Ψ_{I}, Ψ_{D}	delivery rate of intra-coded and incremental frames
$\Psi_{\mathrm{D\|I}}$	conditional delivery rate of incremental frames
PSNR analytical model	
$\mathcal{D}_e$, $\mathcal{D}_u$	Distortion terms (from encoding and decoding)
$\mathcal{D}_{\mathrm{I}}$, $\mathcal{D}_{\mathrm{D}}$	Decoding distortion terms on I and D packets
σ_u^2, σ_e^2	error variance and distortion value from decoding
$\sigma_P^2[t]$	distortion of the propagated error after t frames
$H(\omega, t)$	frequency response of the decoding filters
$\Phi_{uu}(\omega)$	power spectral density of u
γ	leakage coefficient
α	global system parameter

The hybrid ARQ scheme stems from combining retransmissions and FEC. We argue, however, that the structure of the video content is not suitable for applying such techniques without differentiating I and D packets. I packets are more important and also reasonably shorter than D packets; thus, it makes sense to protect them with stronger error control mechanisms. We also propose to limit the retransmissions to these packets, to cause only a limited delay increase. We assume that retransmissions follow a Selective Repeat (SR) approach, which is the most efficient among classic ARQ implementations [11], [13]. In the following, we outline assumptions and notations adopted in the rest of the paper, also reported for ease of reference in Table I.

A. Markov models for packet generation and channel

For analytical tractability, we assume a discrete time axis, which is not restrictive (one can simply increase the sampling frequency to achieve a finer representation). A suitable choice of the sampling step corresponds to the coherence time of the channel, so as to treat the channel quality as constant within a time slot. Both I and D packets span over multiple time slots; actually, packets can be fragmented into multiple transmission units, but these fragments will be transmitted in sequence over adjacent blocks of time slots; thus, we can focus on the time slot as our time scale unit. We assume that the size of an I packet can be between Λ_0 and Λ_1 time slots, whereas that of a D packet can be between Δ_0 and Δ_1 time slots. The specific size of every packet is generated according to a process with memory; for both I and D packets, it is correlated to the size of the last generated I packet. We write $\lambda_j(t|s)$, with $j \in \{\text{I, D}\}$, to denote the conditional probability[1] that the next generated packet of type j is t slots long, given that the last generated I packet is s slots long. This infers a Markov representation akin to those of [9]. According to the criteria proposed there, a suitable Markov model should have more than 2 states; our model has $\Lambda_1 - \Lambda_0 + 1$ states, i.e., easily more than 2.

The wireless channel can also be modeled via Markov chains [10]. This is a widely employed method which both is easy to tune and also accounts for error correlation in wireless environments. The idea is to define a set $\mathcal{X}$ of states, each associated with a different channel error probability, and a transition matrix $\mathbf{P} = (p_{ij})$, where $i, j \in \mathcal{X}$, collecting all transition probabilities from state i to j. Channel correlation, which is a very important issue according to [11], can be taken into account by properly setting $\mathbf{P}$. Note that the Markov state of the channel is regarded as an external element. In a more advanced perspective, which is a possible extension of the present paper, this state can even be used to trigger advanced modulation and rate adaptation mechanisms, so as to generate a cross-layer optimization which would be possible in the context of video over 3G/4G wireless systems [4].

The transitions of the Markov channel are set with a time granularity identical to the time axis. As discussed above, one time slot corresponds to the coherence time of the wireless channel, i.e., the channel behavior within the same time slot is uniform; thus, we can treat each time slot as "correct" or "erroneous." In the following, we assume that errors correspond to erasures, so that the content transmitted over erroneous slots is simply lost, but the receiver is informed of which pieces of information are missing. Again, this is simply one of the possible assumptions about this matter, and other choices can be made as well, resulting in an entirely similar approach; in particular, whenever we are able to correct K erasures, we could correct $\lfloor K/2 \rfloor$ errors instead.

Rather than tracking all channel transitions, we just count how many erroneous slots fall within each packet duration. To preserve the memory of the Markov process, we need to keep record of the channel state in the *last* slot. It is immediate to compute the function $\psi(k, n|j, t)$, for $j, k \in \mathcal{X}$, n, t integers with $0 \leq n \leq t$, which is the probability that, given that t transitions of the channel start from state j, the end state after them is k and exactly n slots out of t are erroneous. This is a well-known function that can be derived in recursive [7] or close [19] form.

[1] Although the proper notation would be something like $\lambda^{(j)}_{T|S}(t|s)$, we prefer to avoid such cumbersome symbols for the sake of readability.

B. Error level

To represent packet transmissions and error control techniques in an integrated manner, we introduce the concept of *error level* [14], which is an extension (or a discrete fuzzification) of the binary representation (correct/erroneous) of packets. Roughly speaking, packets may be described as "partly" erroneous (e.g., "30% erroneous" or "90% erroneous"). Formally, a packet can have an error level chosen among $L+1$ integer values, from 0 (which denotes a correct packet) to L (an entirely erroneous one).

A direct way to obtain the error level is to count the erroneous slots of a packet and scale this number between 0 and L. Thus, a packet of size equal to x slots has error level ℓ if its erroneous slots are between $\lfloor \ell(x+1)/(L+1) \rfloor$ and $\lfloor (\ell+1)(x+1)/(L+1) \rfloor - 1$. Then, we define $\varphi(k, \ell|j, t)$ as the probability that a packet of length t slots starting transmission when the channel is in state j has an error level equal to ℓ and the state of the last slot is k. We derive

$$\varphi(k, \ell|j, t) = \sum_{n=n_1}^{n_2} \psi(k, n|j, t) \ , \qquad (1)$$

$$\text{where: } n_1 = \left\lfloor \frac{\ell(x+1)}{L+1} \right\rfloor, \ n_2 = \left\lfloor \frac{(\ell+1)(x+1)}{L+1} \right\rfloor - 1$$

The error level can be used to determine whether a packet is correctly received. In particular, level 0 means that the packet is entirely error-free (up to the quantization given by the error level); in this case, we are surely able to decode it. However, if FEC capabilities are introduced, some errors can be corrected, and therefore some information content can be acknowledged, even if the error level is larger than 0. To describe this, we introduce an *error correction threshold*, between 0 and L. This value corresponds to the relative amount of redundancy contained in each packet, referred to the range $[0, L]$. Thus, a threshold equal to 0 means that FEC is not used; the larger the threshold, the more powerful the error-correction capabilities. We use two separate thresholds, written as θ_{I} and θ_{D}, for I and D packets, respectively, since the value (and also the role) of the threshold are in general different for I and D packets.

An I packet represents a single frame of the GoP (an I frame), which can be either correct or erroneous according to its error level; this information is sent back as an either positive or negative acknowledgement. In the latter case, if a scheme such as plain ARQ or Type I hybrid ARQ is adopted, which does not make use of incremental redundancy, the packet is retransmitted and an entirely new error level is assigned based on the channel conditions. If a more advanced Type II hybrid ARQ scheme is used, another version of the same information content is sent over the channel, so as to enable packet combining at the receiver.

In the latter case, the effect of retransmissions is to *decrease* the error level from the previous value, which in this manner may be correctly acknowledged. When the involved codes are linear, the correction rules can be translated into linear relationships [20]. Intuitively, if a first transmission and a subsequent retransmission are “70% erroneous” and “20% erroneous,” respectively (and assuming only erasures are involved), the receiver is able to extrapolate the whole information content from the combination of the two packets. Had the retransmission, in the example above, been “50% erroneous,” the receiver would have been unable to fully decode the data. Formally, if two transmissions of the same information content have error levels equal to ℓ and ℓ', respectively, i.e., a fraction of their symbols equal to $(L-\ell)/L$ and $(L-\ell')/L$ is correct, the overall error level of their combination is $[\ell+\ell'-L]^+$, where $[x]^+$ means $\max(0,x)$, as the combination contains a correct fraction equal to $(2L-\ell-\ell')/L$ of a packet. If we were not focusing on erasures, entirely similar equations could have been written, roughly speaking, by reducing the error correction threshold by a factor 2. Also non-linear codes, such as Turbo codes [21], or Low-Density Parity Check (LDPC) and Raptor codes [22] can be treated similarly, by invoking the linearization described in [23]. The important consequence here is that, regardless of the code and the kind of errors for I packets, the ultimate result will still be binary, i.e., an I packet is either correct or not.

If I packets include redundancy, due to FEC or hybrid ARQ mechanisms, they may be acknowledged even when their error level is non-zero. If the amount of redundancy of an I packet is θ_{I}/L, the packet is correctly decoded if the error level ℓ does not exceed θ_{I}. We can regard this from a different perspective, by looking at FEC as a means to decrease the error level of a packet: if the error level is equal to ℓ, FEC decoding is successful if $[\ell-\theta_{\mathrm{I}}]^+$ is equal to 0. When two different transmissions of an I packet with error levels ℓ and ℓ' are combined in a hybrid ARQ fashion, the condition to check becomes whether $[\ell+\ell'-L-2\theta_{\mathrm{I}}]^+$ is equal to 0.

D packets, instead, are an aggregate of many frames, so it is more reasonable to think of the error level as a (rescaled) indicator of how many of them are in error. In other words, if the error level is equal to 0 all involved packets are correct, if the error level is 1 a fraction equal to $1/L$ of them will be in error, and so on. When FEC capabilities are enabled, i.e., the correction threshold θ_{D} is larger than 0, some redundancy is present within the D packet, which we take as equal to θ_{D}/L of the entire data. Thus, if the error level is ℓ, meaning that a fraction ℓ/L of the data is corrupted by erasures, the information content which is still erroneous after FEC decoding is $[\ell-\theta_{\mathrm{D}}]^+/(L-\theta_{\mathrm{D}})$.

III. The Model for Type II Hybrid ARQ

Hybrid ARQ techniques can be distinguished between Type I and Type II [2]. Type I includes additional FEC capabilities but the rationale is that of plain ARQ, i.e., to resend identical copies of not acknowledged packets. Type II exploits incremental redundancy by retransmitting a differently encoded version of the packet; the receiver combines the two packets into a single longer codeword. In this section, we describe the model for Type II hybrid ARQ with SR scheme.

A. Considerations about the feedback

The presence of selective retransmissions requires to follow the outcome of some previously transmitted packets, in addition to that of the packet currently under transmission [12]. As shown in [11], it is sufficient to track those packets whose feedback is still pending. However, the required memory grows exponentially in the number of such packets. In light of this, some researchers consider a simplification, called *ideal* ARQ where positive/negative acknowledgements are immediately known at the transmitter’s side. We argue that this approach is not appropriate for video transmission systems. First of all, we lose any distinction between more efficient SR and other wasteful ARQ schemes, such as Stop-and-Wait (SW). Moreover, the length of video packets is hardly negligible, and so is the processing time at the receiver; ignoring these elements would lead to overly optimistic conclusions. For example, if the transmitter knew that an I packet failed, it would not even transmit its associated D packet, which would be useless; unfortunately, in reality this information is not available. Thus, we assume that, when a feedback message is received, the subsequent packet has already started transmission, without, however, being finished. In this way, we recognize that the feedback is non-instantaneous, still preserving analytical tractability, as there is always one pending packet (the one currently under transmission). In other words, at any time, we track *two* packets, the one for which the feedback is received, referred to as the *packet of interest* and that under transmission, called the *pending packet*. This assumption does not seem in contrast with the video transmission requirements, which impose a timely delivery of the content, therefore compelling the round-trip time to be low. In any event, it would be methodologically simple, though computationally harder, to consider a higher number of pending packets [11]. For the sake of analytical tractability, we assume that I packets are retransmitted at most once. If they are still in error after a retransmission, they are discarded. Not only does this assumption keep the system memory low, but it also appears reasonable as the transmission of video flows has strict time constraints; If needed, it can be relaxed; the model would become more complicated, yet still manageable, as the number of states increases but does not explode.

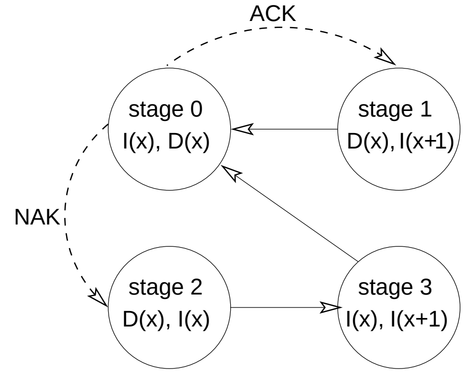


Fig. 1. The macroscopic description of the Type II ARQ system. Below the label of each stage we specify if the packet of interest (left-hand) and the pending packet (right-hand) are of kind I or D.

B. The macroscopic description

We can regard the system as a finite-state machine, represented in Fig. 1, which, each time a feedback for a new packet

$$\pi(0,s_1,s_2,\ell_1,\ell_2,c)=\sum_{x\in\mathcal{X}}\sum_{\ell_0=0}^{L}\left\{\left[\sum_{s_0=\Delta_0}^{\Delta_1}\pi(1,s_0,s_1,\ell_0,\ell_1,x)+\sum_{s_0=\Lambda_0}^{\Lambda_1}\pi(3,s_0,s_1,\ell_0,\ell_1,x)\right]\lambda_{\mathrm{D}}(s_2|s_1)\cdot\varphi(c,\ell_2|x,s_2)\right\} \quad (2)$$

$$\pi(1,s_1,s_2,\ell_1,\ell_2,c)=\sum_{x\in\mathcal{X}}\sum_{\ell_0=0}^{\theta_{\mathrm{I}}}\sum_{s_0=\Lambda_0}^{\Lambda_1}\left\{\pi(0,s_0,s_1,\ell_0,\ell_1,x)\cdot\lambda_{\mathrm{I}}(s_2|s_0)\cdot\varphi(c,\ell_2|x,s_2)\right\} \quad (3)$$

$$\pi(2,s_1,s_2,\ell_1,\ell_2,c)=\sum_{x\in\mathcal{X}}\sum_{\ell_0=\max(\ell_2,\theta_{\mathrm{I}}+1)}^{L}\sum_{s_0=\Lambda_0}^{\Lambda_1}\left\{\pi(0,s_0,s_1,\ell_0,\ell_1,x)\cdot\lambda_{\mathrm{I}}(s_2|s_0)\cdot\sum_{\ell=(L-\ell_0+\ell_2)\,\mathrm{sgn}(\ell_2)}^{L-\ell_0+\ell_2}\varphi(c,\ell|x,s_2)\right\} \quad (4)$$

$$\pi(3,s_1,s_2,\ell_1,\ell_2,c)=\sum_{x\in\mathcal{X}}\sum_{\ell_0=0}^{L}\sum_{s_0=\Delta_0}^{\Delta_1}\left\{\pi(2,s_0,s_1,\ell_0,\ell_1,x)\cdot\lambda_{\mathrm{I}}(s_2|s_1)\cdot\varphi(c,\ell_2|x,s_2)\right\} \quad (5)$$

is received, moves over four macro-states that we call *stages* of the ARQ process.

Stage 0 corresponds to receiving the feedback for an I packet while its associated D packet is pending. The evolution of the system from stage 0 can take either of two paths, according to the packet of interest being acknowledged or not. In the former case, the system evolves to stage 1, where a new I packet is transmitted while the feedback for the D packet, which was previously pending, is received. After this stage, the system is bound to go back to stage 0, where the packet of interest is a new I packet and its associated D packet is under transmission. In the other case, the evolution brings the system to stage 2, where the I packet is retransmitted; now the two packets have the same identifier x, but they are reversed. The system goes then with probability 1 to stage 3, where the packet of interest is this retransmitted I packet, and another newer I packet is pending. Again, after this stage the system transits to stage 0 where the feedback for this latter I packet is received, and its associated D packet is transmitted.

In the following, this representation will be called the *macroscopic description* of the system. The whole system can be seen as a Markov chain, where together with the ARQ stage and the channel condition, represented by variables $\varsigma\in\mathbb{Z}_4=\{0,1,2,3\}$ and $c\in\mathcal{X}$, respectively, we keep memory of the characteristics of the two packets, namely their size and their error level. We assign index 1 to the packet of interest and 2 to the pending packet, and we denote with s_i and ℓ_i the size and the error level, respectively, of the ith packet, with $i\in\{1,2\}$. It is easy to prove that the system is Markov. In fact, the channel and the ARQ stage evolve following a Markov process. The packet generation process is conditioned to the size of the last generated I packet, an information which is always kept in the system; thus, it is also Markov. Finally, the error level depends on the channel evolution and the packet size, according to the function φ introduced in Section II, thus this part is Markov as well. Hence, the system is fully characterized by the 6-tuple $(\varsigma,s_1,s_2,\ell_1,\ell_2,c)$, where the ARQ stage ς is in $\mathbb{Z}_4$, the packet sizes s_j take values from $\min(\Lambda_0,\Delta_0)$ to $\max(\Lambda_1,\Delta_1)$, the error levels ℓ_j are in $\mathbb{Z}_{L+1}=\{0,1,\ldots,L\}$ and $c\in\mathcal{X}$. We denote the corresponding steady-state probability as $\pi(\varsigma,s_1,s_2,\ell_1,\ell_2,c)$.

IV. The Solution of the System Markov Chain

We are now ready to solve the system Markov chain and subsequently compute interesting performance metrics for video transmission over wireless. We also outline how, with effortless changes, the model can describe other techniques, such as Type I hybrid ARQ, plain ARQ and FEC.

A. The steady-state equations

By looking at the macroscopic description, and making use of the channel evolution and packet generation functions, the steady-state probabilities π can be written as satisfying conditions (2)–(5) reported above.[2]

The equations are justified by the following observations. The transitions to stage 0 can come either from 1 or from 3, which justifies the sum of two terms in (2). The transitions to stage 1 must come from stage 0 when the I packet of interest is acknowledged, and this is why (3) contains a sum in ℓ_0 going from 0 to θ_{I}. The transitions to stage 2, evaluated in (4), imply that the I packet was previously not acknowledged (thus ℓ_0 must be higher than θ_{I}). Additionally, the error level ℓ_2 is the result of combining two packets with error level ℓ_0 and ℓ. Thus, the error level ℓ_0 must also be greater than or equal to ℓ_2. We recall that $\mathrm{sgn}(\ell_2)$ denotes the sign of ℓ_2, i.e., it is 0 if $\ell_2=0$ and 1 if $\ell_2>0$. The reason for this term, and for the innermost sum, is that $\ell_2=0$ can be the result of any combination of packets with error levels ℓ_0 and ℓ which produces an entirely correct packet. The last equation, (5), directly follows from the observation that stage 3 is the direct evolution of stage 2. As in this case both packets are of kind I, we use $\lambda_{\mathrm{I}}(s_2|s_1)$, since the length of the latter is correlated to that of the former.

The system of equations written above can be solved by imposing the additional condition that the sum of all π's is 1. As will be shown in the following, deriving the steady-state probabilities enables the computation of many interesting metrics, which can be used to evaluate the system performance. Prior to deriving them, we show that the framework is entirely adaptable to other error control techniques. For example, plain

[2]For the sake of simplicity, we avoid reporting the limiting conditions on the variables in the left-hand side of the equations. They can be derived in a trivial but tedious manner from the stage label. For example, $\pi(0,s_1,s_2,\ell_1,\ell_2,c)$ requires that $\Lambda_0\le s_1\le\Lambda_1$ and $\Delta_0\le s_2\le\Delta_1$ since in stage 0 they refer to an I and a D packet, respectively, and similarly for the other stages.

ARQ, as well as Type I hybrid ARQ (which does not perform any packet combination, thus in our framework is nothing but a variant of plain ARQ), can be obtained by replacing (4) with a simpler version, where there is no mention of the error level ℓ_2 being generated by a packet combination, which is not present in plain ARQ. Thus, instead of (4) we have

$$\pi(2,s_1,s_2,\ell_1,\ell_2,c) = \sum_{x\in\mathcal{X}} \sum_{\ell_0=\theta_{\mathrm{I}}+1}^{L} \sum_{s_0=\Lambda_0}^{\Lambda_1} \Big\{ \pi(0,s_0,s_1,\ell_0,\ell_1,x) \cdot \lambda_{\mathrm{I}}(s_2|s_0) \cdot \varphi(c,\ell_2|x,s_2) \Big\} \quad (6)$$

We obtain FEC in an even simpler manner, by reducing the macroscopic description to an alternation between stages 0 and 1. Since pure FEC does not include any retransmission, the system never transits to stage 2 (and thus, neither to stage 3). The steady-state equations trivially follow.

B. Performance metrics

The average number of slots associated with a transition of the system, denoted as τ, can be computed by summing the steady-state probabilities weighted on the size of the first packet. This corresponds to

$$\tau = \sum_{\varsigma=0}^{3} \sum_{s_1,s_2,\ell_1,\ell_2,c} \left\{ s_1\, \pi(\varsigma,s_1,s_2,\ell_1,\ell_2,c) \right\}, \quad (7)$$

where $\sum_{s_1,s_2,\ell_1,\ell_2,c}$ denotes a sum for s_1, s_2, ℓ_1, ℓ_2, and c over their whole span, i.e., both ℓ_i's go from 0 to L, both s_i's go from $\min(\Lambda_0,\Delta_0)$ to $\max(\Lambda_1,\Delta_1)$, and c spans over the whole set $\mathcal{X}$, respectively.

We can also compute the average delay Ω to transmit a GoP, including retransmissions. To this end, we write $\pi(\varsigma)$ for the sum of all values $\pi(\varsigma,s_1,s_2,\ell_1,\ell_2,c)$ with the same stage ς, and Λ_m and Δ_m for the average sizes of an I and a D packet, respectively. If the I packet is correctly received at its first transmission, Ω is equal to the sum $\Lambda_m + \Delta_m$, otherwise (with probability $\pi(2)$) an additional Λ_m is spent retransmitting the I packet. Thus,

$$\Omega = \Big(1+\pi(2)\Big)\Lambda_m + \Delta_m\,, \quad (8)$$

Another metric we introduce is the average *throughput* Θ, as the amount of correctly delivered data per unit time. As we assumed a saturated source, Θ is the average probability that a unit of traffic gets through; to compute it, we look at the macroscopic description of the hybrid ARQ system. When the system is in stage 0, we have two possibilities. If the I packet of interest is acknowledged, we increase the throughput by its size; moreover, in this case we also count the correct part of the D packet as a further throughput contribution. Else, i.e., if the packet is not acknowledged, we do not count (yet) any contribution. In stage 1 we do not count any contribution, as the D packet, provided that the I packet was correctly received, has already been counted in stage 0. In stage 2 we learn if the I packet which was in error in stage 0 is now acknowledged. If this is the case, we count it as a throughput increase, and so do we for the correct part of the associated D packet (of which we are still keeping memory). Otherwise, as both packets are discarded, we do not count any contribution. As to stage 3, we do not sum any term to the throughput either, because the former I packet has already been taken into account when the system was in stage 2 and the latter (if correct) will be when the system reaches stage 0.

We can formally write down this reasoning by summing over the steady-state probabilities of the macroscopic description being in stage 0 or 2 and dividing by the average transition time τ, i.e.,

$$\Theta = \sum_{s_1,s_2,\ell_2,c} \tau^{-1} \sum_{\ell_1=0}^{\theta_I} \bigg\{ \Big[\pi(0,s_1,s_2,\ell_1,\ell_2,c) + \pi(2,s_2,s_1,\ell_2,\ell_1,c) \Big] \Big[s_1 + s_2 \Big(1 - \frac{[\ell_2-\theta_{\mathrm{D}}]^+}{L} \Big) \Big] \bigg\}, \quad (9)$$

where s_1 and s_2 are reversed between the terms for stages 0 and 2 since I and D packets are in inverse order; again $\sum_{s_1,s_2,\ell_2,c}$ means to sum across the whole span of the indices.

In a similar manner we can compute the *goodput*, denoted as Υ and defined as the actual information content (rather than the data including redundancy symbols as was for the throughput) successfully transmitted per time unit. That is, we weigh any successfully transmitted packet by its code rate. We can proceed similarly to the above derivation of (9) and write

$$\Upsilon = \sum_{s_1,s_2,\ell_2,c} \tau^{-1} \sum_{\ell_1=0}^{\theta_I} \bigg\{ \Big[\pi(0,s_1,s_2,\ell_1,\ell_2,c) + \pi(2,s_2,s_1,\ell_2,\ell_1,c) \Big] \Big[s_1 \Big(1-\frac{\theta_{\mathrm{I}}}{L}\Big) + s_2 \Big(1 - \frac{[\ell_2-\theta_{\mathrm{D}}]^+}{L-\theta_{\mathrm{D}}} \Big) \Big] \bigg\} \quad (10)$$

This metric computes the correctly delivered information content bit-wise (or better, slot-wise). However, as the video content consists of incrementally encoded frames, it may be sensible to estimate the probability that an entire frame is acknowledged, called Ψ_{I} for I frames and Ψ_{D} for frames belonging to a D packet (recall that each D packet comprises multiple frames). To derive these probabilities, we proceed in a similar manner as was done for the other metrics, but we do not divide by τ, as we are interested in the probability of correct delivery, not how long it takes.[3] I packets can be acknowledged at their first or second transmission according to the ARQ stage being 0 or 2, respectively, with corresponding probabilities

$$\Psi_{\mathrm{I}1} = \sum_{s_1,s_2,\ell_2,c} \big(\pi(0)\big)^{-1} \sum_{\ell_1=0}^{\theta_I} \pi(0,s_1,s_2,\ell_1,\ell_2,c)\,,$$
$$\Psi_{\mathrm{I}2} = \sum_{s_1,s_2,\ell_1,c} \big(\pi(2)\big)^{-1} \sum_{\ell_2=0}^{\theta_I} \pi(2,s_1,s_2,\ell_1,\ell_2,c)\,. \quad (11)$$

Therefore, the probability Ψ_{I} can be computed as $\Psi_{\mathrm{I}1} + (1-\Psi_{\mathrm{I}1})\Psi_{\mathrm{I}2}$.

[3]Indeed, it could be taken into account that increasing the amount of time required to deliver a GoP decreases the rate and therefore introduces additional distortion at the encoder. However, this phenomenon is already characterized by the evaluation of the goodput.

We proceed analogously for incremental frames, recalling that they need the correct decoding of their I packet of reference to be delivered. We also assume that the frame errors within a D packet are uniformly distributed, so that if the fraction of erasures in a D packet is Y, then $(1-Y)$ of its frames are correct. The joint probability of correct reception of both a frame within a D packet and its I packet of reference, after one transmission attempt for the latter, is

$$\Psi_{\mathrm{D1}} = \sum_{s_1,s_2,\ell_2,c} \sum_{\ell_1=0}^{\theta_I} \frac{\pi(0,s_1,s_2,\ell_1,\ell_2,c)[L-\max(\ell_2,\theta_{\mathrm{D}})]}{\pi(0)(L-\theta_{\mathrm{D}})}$$

whereas the corresponding value when the I packet is at its second attempt is

$$\Psi_{\mathrm{D2}} = \sum_{s_1,s_2,\ell_1,c} \sum_{\ell_2=0}^{\theta_I} \frac{\pi(2,s_1,s_2,\ell_1,\ell_2,c)[L-\max(\ell_1,\theta_{\mathrm{D}})]}{\pi(2)(L-\theta_{\mathrm{D}})}$$

In the end, we have $\Psi_{\mathrm{D}} = \Psi_{\mathrm{D1}} + (1-\Psi_{\mathrm{I1}})\Psi_{\mathrm{D2}}$, where it is taken into account that a D packet can be decoded only when its related I packet is correct. Thus, Ψ_{D} is a joint probability. The conditional probability that a D packet is correctly decoded, given that the related I packet is correct, is $\Psi_{\mathrm{D|I}} = \Psi_{\mathrm{D}}/\Psi_{\mathrm{I}}$.

C. Analytical evaluation of PSNR

A more detailed quality evaluation for video content can be performed by evaluating further effects due to video coding and decoding, such as error propagation and concealment [24]. Our goal here is to derive an evaluation metric, i.e., the Peak Signal to Noise Ratio (PSNR), which is more commonly adopted in performance evaluation studies about video quality than those reported before (which are instead usual in ARQ studies). Therefore, we compute the Mean Square Error (MSE) of the video sequence averaged over all frames, and from this value we evaluate the PSNR defined as $PSNR = 10\log_{10}(255^2/MSE)$.

The framework that we use for the evaluation is a direct extension of [7]. We briefly sketch it and then discuss how it applies to our evaluations. In [7], the MSE of the received video is seen as the result of summing multiple distortion values, assumed to be uncorrelated, which is valid in most practical cases. Hence, we write $MSE = \mathcal{D}_e + \mathcal{D}_u$, where $\mathcal{D}_e$ is the distortion introduced by the encoder and $\mathcal{D}_u$ is induced at the decoder by residual errors, i.e., those due to the channel and which the error control techniques (ARQ and/or FEC) were not able to solve. While [7] also evaluates $\mathcal{D}_e$ analytically, we only focus on $\mathcal{D}_u$ and assume $\mathcal{D}_e$ as constant, as it does not depend on the error control technique applied.

A background assumption made in [7] to evaluate $\mathcal{D}_u$ is that pixel errors caused by wrongly decoded frames are Gaussian. This works as a reasonable approach for the general case of unknown video; furthermore, it has been extensively validated by simulation, e.g., in [16], and also confirmed by our evaluations. Moreover, it can be assumed that further manipulations of the received signal performed by the codec are linear and time-invariant, and thus can be represented through the frequency responses of some filters. If this filtering is applied iteratively, based on the central limit theorem, one can expect that the impulse response of the filter also becomes Gaussian after a sufficiently large number of iterations [7].

Denote with u the error caused by channel impairments on the pixel values of any decoded frame. In our specific case, as we treated channel errors as erasures, u follows from the fact that missing frames are replaced by duplicating the last correctly received frame [16]. Following [7], and per the above discussion, u is taken Gaussian with zero mean and variance σ_u^2; this last value depends on the frame error rate, as will be explained in the following. First, denote with F the number of frames within a GoP. That is, the "intra" rate is $1/F$ as every F frames, one is intra-coded. Due to incremental encoding, errors propagate within a GoP, so that additional distortion is caused on $F-1$ frames. In practical systems, errors propagate even beyond a single GoP, but this effect is negligible in practice [7]. The square propagated error at time t, for $0 \le t < F$, denoted as $\sigma_P^2[t]$, is

$$\sigma_P^2[t] = \frac{1}{2\pi}\int_{-\pi}^{+\pi} |H(\omega,t)|^2\,\Phi_{uu}(\omega)\,\mathrm{d}\omega\,, \tag{12}$$

where $H(\omega,t)$ is a suitable transfer function and $\Phi_{uu}(\omega)$ is the power spectral density of u.

In [16] Gaussian approximations are taken for both the power spectral density $\Phi_{uu}(\omega)$, which has energy σ_u^2, and the magnitude of the transfer function in the baseband. The related parameters can be specified according to the properties of the decoder; e.g., we write $|H(\omega,t)| = e^{-t\,\omega^2\sigma_f^2/2}$, where the value σ_f can be taken equal to 0.5 in the case of linear interpolation (as commonly used for half-pel motion compensation). However, these values *do not depend* on the frame error rate; they can be simply derived a posteriori as shown next. Plugging them in (12), after some manipulations, we obtain [7]

$$\sigma_P^2[t] = \sigma_u^2\,\frac{1-tF^{-1}}{1+\gamma t} \tag{13}$$

where the leakage factor $\gamma\in[0,1]$ depends on motion compensation techniques. Thus, we compute $\mathcal{D}_u$ as

$$\mathcal{D}_u = \sigma_u^2 \sum_{t=0}^{F-1} \frac{1-tF^{-1}}{1+\gamma t} = \sigma_u^2\alpha \tag{14}$$

$$\text{where: } \alpha = \sum_{t=0}^{F-1} \frac{1-tF^{-1}}{1+\gamma t}\,.$$

The summarizing term α, as discussed above, is not influenced by the frame error probability. For the kind of analysis we are concerned with, the only relevant part is the variance of the error σ_u^2, which instead clearly depends on the channel error probability. In [7], which assumes a homogeneous frame error rate p_{err}, this dependence is modeled as follows. It is assumed that all frames which are in error have the same variance σ_e^2, thus, as long as p_{err} is limited (e.g., $p_{err} \le 0.1$), one can write $\sigma_u^2 = p_{err}\sigma_e^2$. The exact value of σ_e^2 depends on several implementation issues, such as packetization, resynchronization, and error concealment, as well as the specific encoded video sequence (static videos have low variance, whereas rapidly changing scenes have higher values). However, similarly to α, it does not depend on the channel.

We keep this assumption of a constant variance σ_e^2 but, differently from [7], which used a unique value for the error probability, we can have a more precise evaluation thanks to the fact that our analysis differentiates between I and D packets. Thus, we treat the overall distortion due to the decoding error $\mathcal{D}_u$ as the weighted sum of two terms, $\mathcal{D}_\text{I}$ and $\mathcal{D}_\text{D}$, i.e., $\mathcal{D}_u = \mathcal{D}_\text{I} + (F-1)\mathcal{D}_\text{D}/F$. The former term is due to lost I packets, the latter is determined by *additional* losses of D packets (recall that lost I packets cause the whole GoP to be in error). We follow again a reasoning similar to [7] in assuming that the contributions to the overall distortion can be treated as uncorrelated and thus can be summed. To evaluate each term, we use a variation of (14), as follows. To account for errors on D packets, we take a term $\eta_\text{D} = \Psi_\text{I}(1-\Psi_{\text{D}|\text{I}})/F$ and we multiply it by σ_e^2 to obtain σ_u^2. For errors on I packets instead, we want to use a different approach, as they determine the loss of the entire GoP, and thus a long sequence of errors. This means that they have an even more acute decreasing effect on the PSNR. Thus, we define

$$\eta_\text{I} = (1-\Psi_\text{I})F^{(1-\Psi_\text{I})F} \tag{15}$$

which again is meant as a multiplicative coefficient for σ_e^2. If Ψ_I is high enough, this can be seen as an "effective error probability term." This is a heuristic modification, which is, however, justified by the observation that when we have correlated errors and the average length of a burst of errors is K, the probability of exiting from a burst of errors is $1/K$ [7], [11]. Thus, (15) accounts for an additional burst of errors on the predicted frames, of length F. Collecting these results, we derive $\mathcal{D}_u$ as

$$\mathcal{D}_u = \alpha\sigma_e^2 \frac{(1-\Psi_\text{I})F^{(1-\Psi_\text{I})F} + (F-1)/F\ \Psi_\text{I}(1-\Psi_{\text{D}|\text{I}})}{F} \tag{16}$$

where $\alpha\sigma_e^2$ is a constant term which only depends on external factors and just gives a constant bias in all PSNR measurements. Hence, we adopt the same methodology of [7] and derive it a posteriori.

V. Performance evaluation

We numerically evaluate the framework discussed in previous sections within a specific scenario, to show the ability of our model to derive quantitative insight. Following existing analytical characterizations, we consider proper Markov models for the wireless channel and the video packet source [9]–[11]. We also compare the results with the transmission of a real video trace taken as the reference for producing the analytical model of packet generation. Specifically, we consider the *Foreman* sequence (300 frames in QCIF format), encoded in the MPEG-4 format with the default configuration of the `ffmpeg` encoder [25], using a GoP length of $F = 15$ frames (one intra-frame and 14 predictive frames).

A. Choice of the system parameters

The wireless channel is modeled as a two-state Markov chain, with $\mathcal{X} = \{0, 1\}$, where states 0 and 1 mean that the channel is "good" or "bad", i.e., either correct or erroneous, respectively, with probability 1. This model is the same used by many papers [7], [11] and was chosen because the channel transition matrix $\mathbf{P}$ can be fully characterized by just two parameters, namely $\varepsilon = p_{01}/(p_{10} + p_{01})$ and $B = 1/p_{10}$, which represent the steady-state channel error probability and the average error burst length, respectively. The case of $B = (1-\varepsilon)^{-1}$ corresponds to independent and identically distributed (iid) channel errors. Finally, we consider an error level quantization based on $L + 1 = 6$ levels.

The video source is modeled according to [9], which has been extended to take into account that each I packet maps an intra-frame, and each D packet represents an aggregate of $F-1$ predictive frames. Following [9] and based on realistic values, we take I packets as following a steady-state truncated Gamma distribution, where the size of an I packet is between $\Lambda_0 = 4$ and $\Lambda_1 = 10$ slots. Due to the discrete time axis, the Gamma distribution becomes a negative binomial distribution. Hence, the steady-state probability that an I packet occupies j slots, for $j \in \{\Lambda_0, \dots, \Lambda_1\}$, was computed as a negative binomial distribution of 3 successes, where the success probability is 0.4, as per [9]; the values of the distribution for lengths less than Λ_0 (or greater than Λ_1) are cumulated in the value for Λ_0 (respectively, Λ_1).

The steady-state distribution values are replicated over all the rows of a $\Lambda_1 - \Lambda_0 + 1$–square matrix $\mathbf{Q}$. Thus, every entry q_{ij} with $i, j = 0, \dots, \Lambda_1 - \Lambda_0$, is equal to the steady-state probability that an I packet has length $\Lambda_0 + j$. To introduce correlation among I packets, we follow again [9] and consider a correlation factor ρ (equal to 0.8 in the numerical evaluations). The actual transition matrix of the Markov process generating the length of I packets is the result of a first-order filtering between conserving the same length for the next I packet with probability ρ, and checking the transition on the matrix $\mathbf{Q}$ with probability $1-\rho$. In other words, we can set $\lambda_\text{I}(j+\Lambda_0|i+\Lambda_0) = \widetilde{q}_{ij}$, i.e., as the elements of the matrix $\widetilde{\mathbf{Q}} = (\widetilde{q}_{ij}) = \rho\,\mathbf{I} + (1-\rho)\mathbf{Q}$, where $\mathbf{I}$ is a $(\Lambda_1 - \Lambda_0 + 1)$-sized identity matrix.

D packets are instead generated with length between $\Delta_0 = 8$ and $\Delta_1 = 40$, according to a truncated and sampled Gaussian distribution $\lambda_\text{D}(t_1|t_0)$ with mean equal to $2.8t_0$ and standard deviation equal to $0.5t_0$ slots, where t_0 is the length of the corresponding I packet. The resulting average sizes of I and D packets are $\Lambda_m = 5.337$ and $\Delta_m = 14.943$ slots, respectively. When error correcting codes are employed on the packets, their lengths are increased accordingly; e.g., if a packet is coded with rate R, it becomes $1/R$ times longer.

B. Compared techniques

We consider different error control techniques, classified according to the following taxonomy. Recall that D packets can be protected by FEC, but are not retransmitted. Thus, if the same amount of FEC is used, the actual distinction is made by the technique applied to I packets. If just plain ARQ without any FEC is applied to I packets, we denote the resulting technique as ARQ-0. A variant of this is achieved by introducing FEC in the I packets, but still without applying

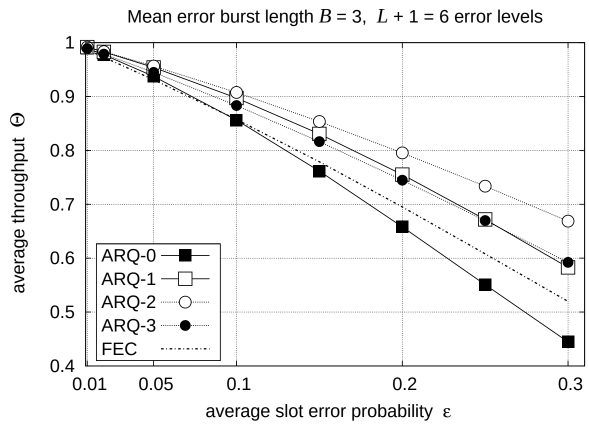


Fig. 2. Throughput Θ versus the average slot error probability ε for $B = 3$.

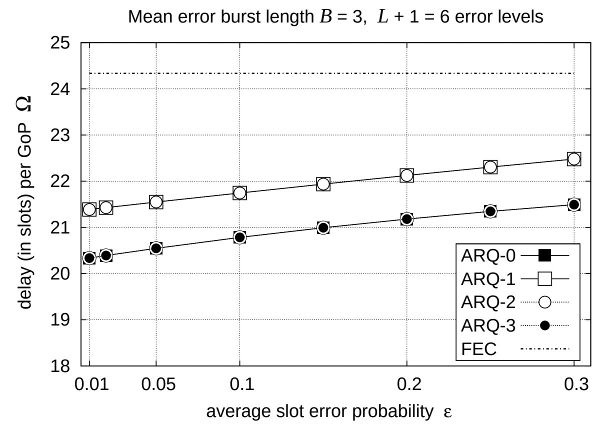


Fig. 4. GoP delay Ω versus the average slot error probability ε for $B = 3$.

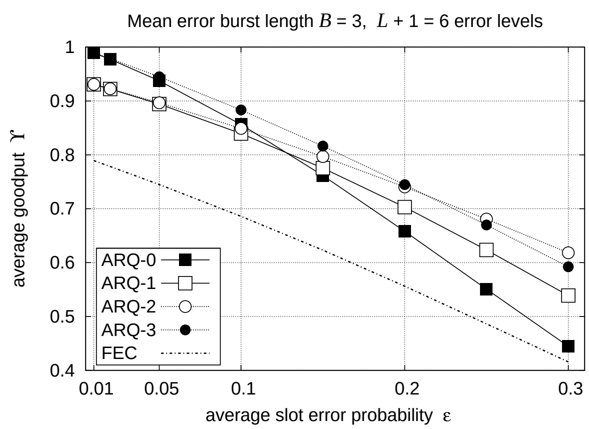


Fig. 3. Goodput Υ versus the average slot error probability ε for $B = 3$.

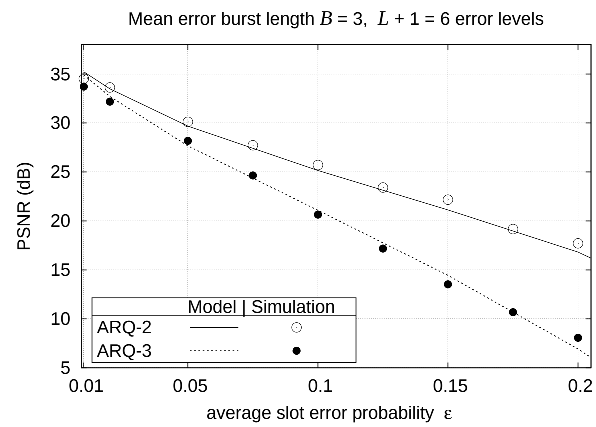


Fig. 5. Packet error rates versus the average slot error probability ε for $B = 3$.

packet combining; thus, the resulting technique (limited to I packets) is a Type I hybrid ARQ, and we denote it as ARQ-1. If Type II hybrid ARQ is adopted, i.e., all packets are protected with coding and I packets are retransmitted with incremental redundancy packet combining, we have what we call ARQ-2. Finally, another technique is introduced as a mixture of ARQ-0 and ARQ-2, i.e., we combine packets in a Type II hybrid ARQ fashion, but the first packet sent does not contain any redundancy. As this technique is sometimes called Type III ARQ [1], we denote it as ARQ-3. Together with these four ARQ techniques we also implement a simple FEC scheme, where no retransmission is performed and the flow is coded without differences between I and D packets.

C. Numerical results

We consider first a scenario with a mildly correlated wireless channel, i.e., with $B = 3$, and variable average error probability. In all the ARQ schemes, no error control is applied to D packets, only I packets are protected by some technique, as discussed above. In ARQ-1 and ARQ-2, the code rate for I packets is $4/5$. The pure FEC scheme, instead, applies an error-correcting code with rate $4/5$ to all packets.

Our analytical framework enables evaluating the throughput Θ, reported in Fig. 2. It can be seen that ARQ-2 achieves the highest throughput; it is also worth noting that ARQ-3 achieves good performance, very similar to that of ARQ-1, without using any FEC mechanism directly, but rather only combining erroneous packets in a hybrid ARQ fashion. This conclusion is confirmed and extended when looking at Fig. 3, where the goodput Υ is reported. Here, the curves of ARQ-0 and ARQ-3, which do not include additional redundancy due to FEC, are the same of Fig. 2, whereas ARQ-1, ARQ-2 and FEC decrease accordingly. In this way it is visible that all ARQ techniques perform better, for what concerns goodput, than pure FEC. The scheme with highest goodput is ARQ-3, but ARQ-2 still offers good performance and actually becomes the best choice when the average error probability increases.

Fig. 4 reports the GoP delay Ω. ARQ-0 and ARQ-3 perform identically, and so do ARQ-1 and ARQ-2, as they have the same macroscopic evolution. Since they do not introduce any redundancy on D packets, they outperform the constant delay achieved by FEC, where every packet is protected with coding and always transmitted once. This result establishes that ARQ-like techniques do not necessarily cause large delays; therefore, it is not true that FEC is the only viable choice for error protection, instead hybrid ARQ techniques with selective retransmissions can achieve an even better delay performance. The ARQ techniques considered here still do not protect D packets; some FEC may be applied to them, which will cause

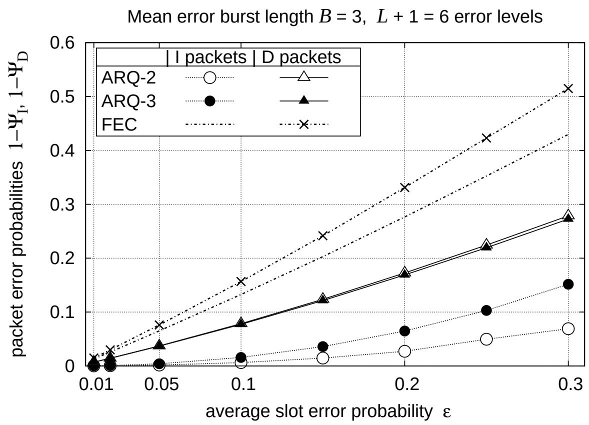


Fig. 6. PSNR versus the average slot error probability ε for $B = 3$.

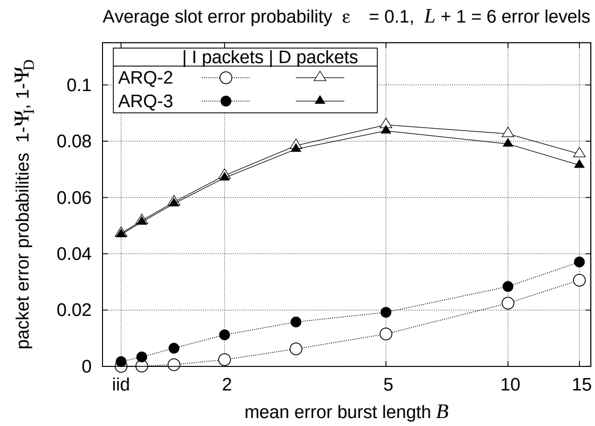


Fig. 8. Packet error rates versus the average burst length B for $\varepsilon = 0.1$.

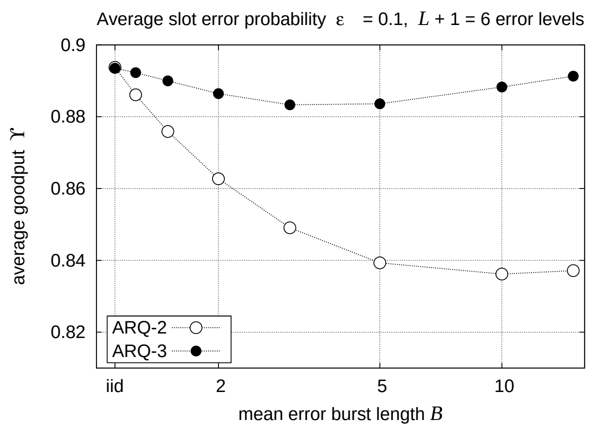


Fig. 7. Goodput Υ versus the average burst length B for $\varepsilon = 0.1$.

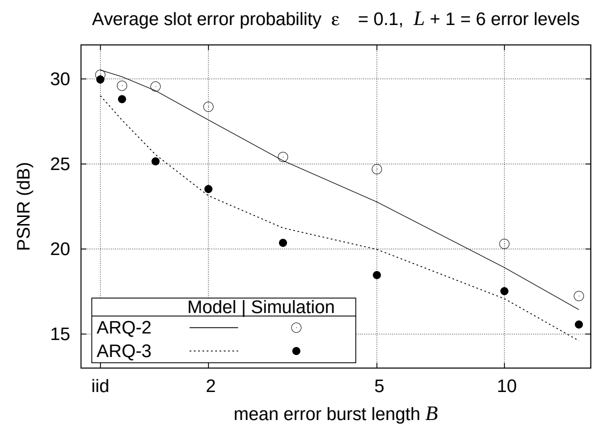


Fig. 9. PSNR versus the average burst length B for $\varepsilon = 0.1$.

the delay to increase accordingly. Nevertheless, there is some margin to protect the incremental frames without raising the delay too much (or at least, to perform comparably with FEC).

In Fig. 5, we also show the terms $1 - \Psi_{\mathrm{I}}$ and $1 - \Psi_{\mathrm{D}}$, i.e., the error rates of I and D packets, respectively. To have a clear representation, we omitted ARQ-0 and ARQ-1; their performance is somewhat intermediate between the other ARQ techniques and pure FEC. For D packets, the curves of ARQ-2 and ARQ-3 are almost identical, in fact they treat the D packets in the same manner. Moreover, ARQ-3 and especially ARQ-2 outperform FEC. This quality evaluation can be translated into a more common measurement of PSNR, which is reported in Fig. 6 and also compared with simulation results for a realistic video transmission. In this latter case, the packetization at the sender, the reconstruction of the video at the receiver and the PSNR calculations were performed using *Evalvid* [26]. The simulated transmission of the video stream was performed on the same Markov channel of the analytical approach, averaged over a large number of realizations. During these simulated runs, the packets which were in error after the application of error control techniques were deleted from the stream, which was subsequently decoded using `ffmpeg`; this procedure also implements additional error concealment capabilities [24]. We limit the plots to ARQ-2 and ARQ-3 as they are the only ones falling within the validity range of the analysis (error rate below, or comparable with, 10%). Indeed, the other techniques fail to obtain a meaningful PSNR due to very high error rates, so the curve will drop to 0 almost immediately. Compared with simulation results, our extension of the analytical PSNR evaluation offers good accuracy.

The impact of channel correlation is investigated in Fig. 7, where we report Υ as a function of B. It is emphasized that error correlation impacts on the performance; the overall performance is significantly changed from the iid case when even a limited value of B is considered. Especially, the application of coding is shown to be less effective than retransmitting the packet (which justifies why ARQ-3 performs better than ARQ-2 apart from the iid case), as the latter technique can better avoid error bursts. Other considerations can be drawn by looking at the error rates reported in Fig. 8. In particular, the error rate of I packets is shown to increase for higher channel correlation. The curves for D packets have instead a maximum around the point where the average burst length is equal to the size of an I packet. Fig. 9 correspondingly shows a decrease of PSNR for higher values of B.

Finally, we evaluate the introduction of FEC on the D packets in Figs. 10 and 11. Here, ARQ-2 is considered, but similar results can be drawn for the other schemes. Besides the case, already shown, with no FEC, we consider two values for θ_{D} which correspond to applying codes to the D packets with

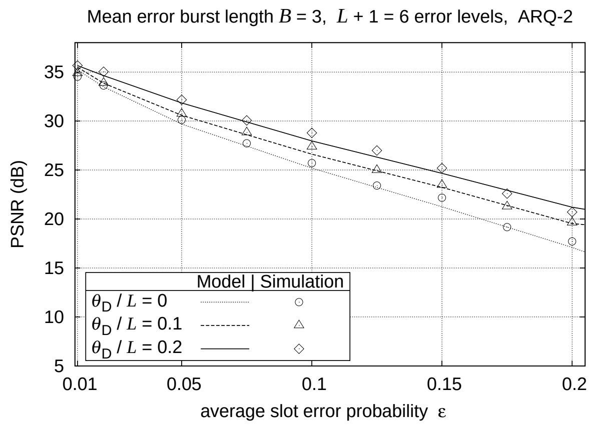


Fig. 10. Goodput Υ as a function of the average slot error probability ε for $B = 3$ and various strengths of FEC on the D packets for ARQ-2.

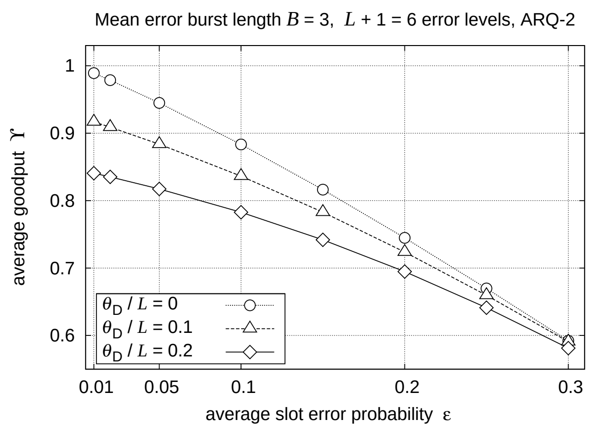


Fig. 11. PSNR evaluation as a function of the average slot error probability ε for $B = 3$ and various strengths of FEC on the D packets for ARQ-2.

rate $9/10$ and $4/5$ (the same as the I packets), respectively. Protecting the incremental content with more FEC increases the quality (Fig. 11) but at the same time decreases the goodput (Fig. 10). A trade-off is present, which can be captured by our model in an entirely modular manner, by simply tuning the proper parameter. These results can be put in relationship with Figs. 3 and 6, respectively, where the application of FEC to I packets is considered instead (thus resulting in the comparison between ARQ-2 and ARQ-3). It is worth noting that while Fig. 6 showed a significant quality improvement by increasing the protection for I packets, the effect of error control techniques on D packets is shown here to be marginal.

VI. Conclusions

We developed an entirely analytical tool based on Markov chains to study error control techniques applied to video transmission over wireless. The transmission process is modeled by means of a Markov chain also including the channel state and the packet generation process. Our general theoretical approach is meant to capture the main characteristics of video transmission over wireless from an analytical standpoint.

Our proposed approach can serve first of all as an evaluation benchmark; for instance, in this paper we compared plain ARQ, FEC and hybrid error control schemes in terms of throughput, goodput, delay, and PSNR, and these assessments were also verified by means of simulation. We were able to disprove the misconception that ARQ-like techniques cannot be used for video transmission over wireless due to excessive delays. Actually, this was proven not to be true if retransmissions are limited to the intra-coded packets, which have a key role in the video transmission. When such packets are protected via ARQ with a limited number of transmission attempts, the resulting hybrid ARQ schemes were shown to better counteract the wireless channel impairments and improve the overall perceived video quality. The presented analytical framework is open to many applications, from the formulation of general frameworks for the optimization of modulation and coding to the evaluation of inherent tradeoffs and the derivation of useful guidelines for video transmission systems over wireless channels.

References

[1] S. Kallel, "Complementary punctured convolutional (CPC) codes and their applications," *IEEE Trans. Commun.*, vol. 43, no. 6, pp. 2005–2009, Jun. 1995.

[2] S. Lin and P. Yu, "A hybrid ARQ scheme with parity retransmission for error control of satellite channels," *IEEE Trans. Commun.*, vol. 30, no. 7, part 2, pp. 1701–1719, Jul. 1982.

[3] X. Zhang and Q. Du, "Adaptive low-complexity erasure-correcting code-based protocols for QoS-driven mobile multicast services over wireless networks," *IEEE Trans. Veh. Technol.*, vol. 55, no. 5, pp. 1633–1647, Sep. 2006.

[4] F. Zhai, Y. Eisemberg, T. N. Pappas, R. Berry, and A. K. Katsaggelos, "Rate-distortion optimized hybrid error control for real-time packetized video transmission," *IEEE Trans. Image Process.*, vol. 15, no. 1, pp. 40–51, Jan. 2006.

[5] G. K. Wallace, "The JPEG still picture compression standard," *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, Apr. 1991.

[6] F. Fitzek and M. Reisslein, "MPEG4 and H.263 video traces for network performance evaluation," *IEEE Netw.*, vol. 15, no. 6, pp. 40–54, 2001.

[7] K. Stuhlmüller, N. Färber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 1012–1030, Jun. 2000.

[8] Y. Pourmohammadi Fallah, H. Mansour, S. Khan, P. Nasiopoulos, and H. Alnuweiri, "A link adaptation scheme for efficient transmission of H.264 scalable video over multirate WLANs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 7, pp. 875–887, Jul. 2008.

[9] D. Heyman, A. Tabatabai, and T. Lakshman, "Statistical analysis and simulation study of video teleconference traffic in ATM networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, no. 1, pp. 49–59, Mar. 1992.

[10] Q. Zhang and S. A. Kassam, "Finite-state Markov model for Rayleigh fading channels," *IEEE Trans. Commun.*, vol. 47, no. 11, pp. 1688–1692, Nov. 1999.

[11] M. Rossi, L. Badia, and M. Zorzi, "Exact statistics of ARQ packet delivery delay over Markov channels with finite round-trip delay," *IEEE Trans. Wireless Commun.*, vol. 4, no. 4, pp. 1858–1868, Jul. 2005.

[12] Z. Rosberg and M. Sidi, "Selective-Repeat ARQ: the joint distribution of the transmitter and the receiver resequencing buffer occupancies," *IEEE Trans. Commun.*, vol. 38, no. 9, pp. 1430–1438, 1990.

[13] M. E. Anagnostou and E. N. Protonotarios, "Performance analysis of the Selective-Repeat ARQ protocol," *IEEE Trans. Commun.*, vol. 34, no. 2, pp. 127–135, 1986.

[14] L. Badia, M. Levorato, and M. Zorzi, "Markov analysis of selective repeat type II hybrid ARQ using block codes," *IEEE Trans. Commun.*, vol. 56, no. 9, pp. 1434–1441, Sep. 2008.

[15] H. Liu and M. E. Zarki, "Performance of H.263 video transmission over wireless channels using hybrid ARQ," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 9, pp. 1775–1786, Dec. 1997.

[16] B. Girod and N. Färber, "Feedback-based error control for mobile video transmission," *Proc. IEEE*, vol. 87, no. 10, pp. 1707–1723, Oct. 1999.

[17] L. Badia, "On the effect of feedback errors in Markov models for SR ARQ packet delays," in *Proceedings IEEE Globecom*, Honolulu, HI, Dec. 2009.

[18] J. Tang and X. Zhang, "QoS–driven power allocation over parallel fading channels with imperfect channel estimations in wireless networks," in *Proceedings IEEE Infocom*, Anchorage, AK, May 2007, pp. 62–70.

[19] M. Zorzi and R. Rao, "Lateness probability of a retransmission scheme for error control on a two-state Markov channel," *IEEE Trans. Commun.*, vol. 47, pp. 1537–1548, 1999.

[20] S. B. Wicker, *Error control systems for digital communication and storage*. Englewood Cliffs, NJ: Prentice-Hall, 1994.

[21] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo codes," *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261–1271, Oct. 1996.

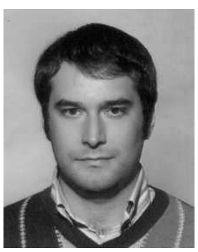
[22] E. Soljanin, N. Varnica, and P. Whiting, "Incremental redundancy hybrid ARQ with LDPC and Raptor codes," *submitted to IEEE Trans. Inform. Theory*, Sep. 2005.

[23] L. Badia, M. Levorato, and M. Zorzi, "A channel representation method for the study of hybrid retransmission-based error control," *IEEE Trans. Commun.*, vol. 57, no. 7, pp. 1959–1971, Jul. 2009.

[24] P. Salama, N. B. Shroff, and E. J. Delp, "Error concealment in MPEG video streams over ATM networks," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 1129–1144, Jun. 2000.

[25] F. Bellard. FF-MPEG: A complete solution for recording, converting and streaming audio and video. Accessed on Jan. 2009. [Online]. Available: http://ffmpeg.org

[26] J. Klaue, B. Rathke, and A. Wolisz, "Evalvid–a framework for video transmission and quality evaluation," ser. Lecture notes in computer science. Berlin/Heidelberg: Springer, 2003, vol. 2794, pp. 255–272.



Leonardo Badia (S'02, M'04) was born in Ferrara, Italy, in 1977. He received the Laurea Degree (with honors) in Electrical Engineering and the Ph.D. in Information Engineering from the University of Ferrara, Italy, in 2000 and 2004, respectively. During 2002 and 2003 he was on leave at the Radio System Technology Labs (now Wireless@KTH), Royal Institute of Technology of Stockholm, Sweden. After having been with the Engineering Department of the Università di Ferrara, he joined in 2006 the "Institutions Markets Technologies" (IMT) Lucca Institute for Advanced Studies, in Lucca, Italy, where he is currently an Assistant Professor. His research interests include protocol design for multihop networks, cross-layer optimization of wireless communication, transmission protocol modeling, and applications of game theory to radio resource management. Dr. Badia serves on the Editorial Board of the Wiley Journal of Wireless Communications and Mobile Computing, and is an active reviewer for several IEEE periodicals in the communication area.

Nicola Baldo (S'07) was born in Rovigo, Italy, in 1981. He received his Laurea (BE) and Laurea Specialistica (ME) degree in Telecommunications Engineering in 2003 and 2005, respectively, from the University of Ferrara, Italy, and the PhD in Information Engineering from the University of Padova, Italy, in 2009. In summer 2003, he was an internship student at Ericsson Eurolab Deutschland, Aachen, Germany, working on Wireless Multimedia Communications. In 2005, he was on leave at the STMicroelectronics Advanced System Technology group, Agrate Brianza, MB, Italy, working on Cross-layer Optimization for VoIP over Wireless LAN. In 2008, he was on leave at the California Institute for Telecommunications and Information Technology (Calit2), University of California, San Diego, USA, working on Cognitive Networks and Software Defined Radio. Since February 2009, he has been a post-doc researcher at the Centre Tecnològic de Telecomunicacions de Catalunya, Barcelona, Spain. His research interests include cognitive radio and networks, cross-layer optimization, multimedia communications and network simulation tools.

Marco Levorato (S'06, M'09) was born in Venice, Italy, in 1980. He obtained both the BE (Electronics and Telecommunications Engineering) and the ME (Telecommunications Engineering) *summa cum laude* from the University of Ferrara, Italy, in 2002 and 2005, respectively. During 2005 he held a fellowship at the University of Padova, Italy. In 2009, he received a PhD in Information Engineering from the University of Padova. During 2008 he was on leave at the University of Southern California, Los Angeles, United States. Currently he is a post-doc researcher at the University of Padova. His research interests include cooperative communications, design of ad hoc networks with multiuser detection and analysis of Hybrid ARQ techniques.

Michele Zorzi (S'89, M'95, SM'98, F'07) was born in Venice, Italy, in 1966. He received the Laurea Degree and the Ph.D. in Electrical Engineering from the University of Padova, Italy, in 1990 and 1994, respectively. During the Academic Year 1992/93, he was on leave at the University of California, San Diego (UCSD), attending graduate courses and doing research on multiple access in mobile radio networks. In 1993, he joined the faculty of the Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy. After spending three years with the Center for Wireless Communications at UCSD, in 1998 he joined the School of Engineering of the University of Ferrara, Italy, where he became a Professor in 2000. Since November 2003 he has been on the faculty at the Information Engineering Department of the University of Padova. His present research interests include performance evaluation in mobile communications systems, random access in mobile radio networks, ad hoc and sensor networks, energy constrained communications protocols, and underwater communications and networking. Dr. Zorzi was the Editor-In-Chief of the IEEE Wireless Communications Magazine from 2003 to 2005, is currently the Editor-In-Chief of the IEEE Transactions on Communications, and serves on the Steering Committee of the IEEE Transactions on Mobile Computing, and on the Editorial Boards of the Wiley Journal of Wireless Communications and Mobile Computing and the ACM/URSI/Kluwer Journal of Wireless Networks. He was also guest editor for special issues in the IEEE Personal Communications Magazine ("Energy Management in Personal Communications Systems") and the IEEE Journal on Selected Areas in Communications ("Multi-media Network Radios" and "Underwater wireless communications and networking"). He is a Member-at-Large of the Board of Governors of the IEEE Communications Society.