A Markov Analysis of Automatic Repeat Request for Video Traffic Transmission

Leonardo Badia and Anna V. Guglielmi
Dept. of Information Engineering, University of Padova via Gradenigo 6B, 35131 Padova, Italy
email: {badia, guglielmi}@dei.unipd.it

***Abstract*—This paper presents a study of the automatic repeat request (ARQ) technique applied to the transmission of multimedia traffic, e.g., video content. In the literature, retransmission-based techniques are usually investigated by means of queueing theory and assuming a homogeneous flow of identical packets, which are sent and possibly retransmitted all in the same way. However, multimedia packets are the result of an incremental encoding that leverages spatial and temporal redundancy, which is naturally present in the raw data. As a result, the flow is inherently made of packets with different roles, which should also be treated differently by the ARQ mechanism. Thus, we assume that different levels of error protection are applied, and also we model the decoding process at the receiver as accounting for a dependence relationship among the packets. Moreover, since error correlation has a strong impact on the performance, we consider a transmission over a Markov channel where we tune not only the error probability but also the average error burst size. This enables the derivation of several performance metrics in an entirely analytical manner via Markov analysis. Finally, some numerical results are explored and possible applications on the development of guidelines for multimedia transmission are discussed.**

***Index Terms*—Automatic repeat request, Markov processes, error analysis, queueing analysis, multimedia communications.**

I. Introduction

Automatic repeat request (ARQ) is an error control technique, which exploits feedback information about faulty reception of some packets to trigger their retransmissions [1], [2]. The selective repeat (SR) version of the ARQ mechanism is the most efficient basic implementation that specifically retransmits only those packets who are reported to be in error [3]. In the last few years [4]–[9], there has been a renewed interest for ARQ, also including hybrid ARQ techniques, where plain retransmission schemes are coupled with forward error connection (FEC). ARQ-based error control is used in high-quality multimedia applications [10] and included in the evolution of the Universal Mobile Telecommunications System (UMTS) towards high speed packet access (HSPA) and the so-called Long Term Evolution (LTE) [11].

Since multimedia contents, and in particular video, are expected to be the dominant component of traffic over future generation networks, it is important to understand their behavior and analytically characterize their performance. Unfortunately, most investigations about multimedia transmission just resort to simulation, without strong analytical support for systematic evaluations. We believe that the reason for multimedia content being difficult to characterize in an analytical manner is related to the incremental nature of source coding that is applied to such flows. Video flows are normally obtained through incremental encoders such as those defined by Moving Picture Experts Group (MPEG) [12]. As a result, different packets have different roles within the video flow and therefore the performance of an error correction scheme, such as ARQ, cannot be simply described by a residual error probability. From the networking point-of-view, it matters whether the missing packets belong to an independently encoded, rather than an incremental frame [6].

At the same time, it is also unclear how to apply ARQ-like techniques to such flows. An objection that is often raised against retransmission-based error control for video is that it would cause the delay to grow, which is not acceptable for real-time content. Actually, another side purpose of the present paper is to disprove such a claim, and to do so with an entirely analytical characterization. In fact, we will show that ARQ can be applied quite easily to video content, if selective retransmissions are carefully chosen.

Thus, the aim of this paper is to propose a mathematical model exploiting discrete-time Markov chains to represent the SR ARQ transmission mechanism for multimedia content. This requires to keep into account interdependences of packets in the source coding [7], and also to evaluate their impact at the receiver's side in case a packet is missing, which will also be differently evaluated depending on the kind of the packet. The main hurdle for ARQ evaluations is that classical investigations, which can also be seen as extensions of queueing theory, do not directly apply as the flows consists of different types of packet. However, by using a synthetic description of the inherent structure of the packet group, that we call *macroscopic representation* [9], we are able to greatly simplify the system analysis.

As a result, the paper poses the following contributions. First, we produce an exact Markov analysis of ARQ applied to video content, which will be used to quantify some networking performance metrics (throughput, number of retransmissions, packet discarding probability). Such a derivation can be extended to any system of choice and can be used as a preliminary testing of video transmission performance.

Second, we investigate the impact on such metrics of channel characteristics. In our analysis, a Markov channel [5] is considered, whose tunable parameters include average packet error rate and average error burst length, which is an indicator of correlation. The analysis can therefore identify the impact of these parameters (especially comparing the effects of channel error rate alone versus correlation among errors)

on the resulting video quality. We remark that the impact of error correlation is especially worth of investigation, although it is rarely addressed in most evaluations.

Finally, we show how ARQ can be implemented for such flows, by identifying a selective retransmission mechanism that not only chooses to retransmit just packets in error, but does so exclusively for the important ones. A sliding-window mechanism is identified to sacrifice incremental packets and replace them with retransmissions of the most important packets which are in error. In this way, not only the performance is improved, but no delay increase is incurred, which is a key advantage for video transmission.

The rest of this paper is organized as follows. In Section II we discuss related works. Section III describes the ARQ model and defines the assumptions that are used to study it. Section IV presents the proposed implementation of ARQ for video streams and its evaluation through the solution of a Markov chain. To this end, different performance metrics are discussed and presented. Section V introduces and discusses some numerical results. Finally, Section VI concludes the paper.

II. Related work

Classical references discussing the performance of retransmission-based error control for multimedia content date back to the end of the last century [13]–[15]. In particular, Liu and El Zarki [13] investigate hybrid ARQ for an H.263 video flow, while Girod and Färber [14] and the further extension by Stuhlmüller et al. in [15] evaluate realistic performance of transmitted videos, but they actually only focus on FEC; with some effort, this analysis could be extended to a basic stop-and-wait ARQ [3].

All these investigations emphasize that the nature of multimedia content does not permit to directly apply standard evaluations taken from queueing theory and protocol analysis. In other words, due to incremental encoding among the packets, the throughput (meant as the amount of data that can be reproduced at the receiver's side) cannot be taken as the correct packet delivery ratio. It is well possible that certain packets are correctly delivered, but still they are useless for the multimedia receiver as the errors on other packets of the flow make it impossible to decode the content.

However, the aforementioned references do not dwell into the details of a full theoretical analysis, mostly due to complexity reasons that the resulting framework would have (at least for the software instruments of that time). From a network performance point of view, the only evaluation is made by means of simulation, while all the approaches presenting a theoretical analysis still consider a homogeneous flow of packets.

Around the 2000s there has also been an evolution of investigations about ARQ exploiting more advanced analytical instruments and also higher computational power [4], [5], [16], [17]. These investigations extend the classic papers about ARQ [2], [3] and obtain the derivation of interesting metrics from an overall performance standpoint. In particular, not only average values but also full statistics can be obtained. Without certain limits, as the computational complexity still grows exponentially, the analysis can be performed for large round-trip times and therefore several pending packets in the ARQ window.

In particular, Cam and Leung in [16] evaluated throughput metrics of ARQ systems and their dependence on channel performance. While the distribution of the buffer occupancy for SR ARQ was already evaluated in a classical contribution by Rosberg and Sidi [18], and there existed also an approximated analysis of the packet delay by Anagnostou and Protonotarios [19], there was no explicit statistical analysis that extended the characterization of ARQ to packet delay, and this was provided by Kim and Krunz in [17].

This contribution is further extended by Rossi et al. in [4], where it is proposed to model SR ARQ through an approach based on discrete-time Markov chains and accounting for the impact of non-instantaneous feedback. To compute the delay statistics, a two-state Markov channel model is used and a larger Markov chain is built by including past history of packets pending acknowledgment. A similar approach is also used by Luo et al. [5] to determine the performance of adaptive modulation and coding schemes. However, while all the aforementioned approaches consider SR ARQ, they only focus on homogeneous flows of identical packets and they obtain packet delay statistics, where quality is quantified as fully reliable delivery of all packets, something that would be very hard to guarantee for video flows.

Other contributions around the same years present similar approaches, all sharing the approach of considering the data transmitted through ARQ as a homogeneous flow, where all the packets are identical. Fewer papers characterize instead ARQ applied to multimedia streams, where the packets have different roles (in particular, some are more important than others for decoding the flow). Incidentally, this also requires a re-thinking of the whole retransmission mechanism, motivated by the simple observation that if the packets are of different importance, they should also be handled differently. Classic packet differentiation proposes to adopt different forward error correction levels depending on the packet type [14]. However, this should also be applied to the retransmission counterpart in ARQ or hybrid ARQ. Such a remark was independently raised by Seo et al. in [6] and Zhang and Du in [7]. However, even though these papers propose an application of non-uniform error protection to multimedia flows, they do not provide an analytical characterization of the resulting performance, which is the goal of the present paper.

Finally, another related reference is a previous work [9] by Badia et al., with which the present paper shares some similarities in the idea of exploiting a Markov chain to characterize an ARQ mechanism with different retransmission priorities for the multimedia packets, depending on their role in the encoding. However, there are many notable differences between [9] and the present contribution. First, that analysis focuses on a variable-length frame transmission, which is characterized in terms of quality performance, and results about video-related metrics, such as PSNR, are provided. In this paper, we focus instead on ARQ-related metrics, including throughput, average number of retransmissions, and

packet discarding probability. Even more importantly, [9] only considers an ARQ window containing at most one pending packet. For these reasons, the impact of the retransmission pattern is not investigated there (basically, that analysis is just a stop-and-wait ARQ), differently from our investigations that involves a more efficient SR ARQ. This also opens up a further challenge on how to schedule retransmissions between the packet flow, which is discussed in the next sections.

III. Assumptions and Model of ARQ Transmission

Consider a transmitter-receiver pair that communicate over a lossy (and correlated) channel. The transmitter sends a multimedia content, which is modeled as a flow of packets of different types. In particular, we consider two kinds of packets, dubbed as "type A" and "type B," respectively. The former type of packets includes *independently* encoded packets; for a video transmission they can be considered as (part of) a single frame that can be played even as a stand-alone content. The latter instead correspond to *incrementally* encoded packets, whose decoding depends on other packets of the flow.

For the sake of simplicity, we make the assumptions that (i) all packets are of equal size; (ii) each B-packet is encoded from exactly one A-packet, so that one A-packet serves as the encoding basis for a given number, denoted as $F-1$, of B-packets; (iii) the packets are transmitted in a periodic pattern, in particular, we consider a sequence of one A-packet followed by all the B-packets encoded from it. Thus, the basic transmission element is a *group* of F packets, led by one A packet. This naming is akin to that used in the literature [6], [8] when the analysis is intentionally general. Also, apart from some simplifications (bidirectional encoding is neglected, and so is dependence among incremental packets), the structure is reminiscent of the structure of MPEG videos, and a group of F packets is the analogous of a Group of Pictures (GoP) in MPEG, which can be seen as an I-frame followed by some P-frames (without any B-frame, under the simplification that there is no bidirectional dependence of packet encoding). Actually, a single B-packet in our representation is better seen as a bunch of P-frames from MPEG encoding, since P-frames are usually shorter than I-frames. However, it is worthwhile noting that, according to the analysis reported in [9], the impact of different packet sizes is marginal compared to the interdependence of the encoding. In other words, assuming equally sized A-packets and B-packets does not appear to be a very restrictive assumption.

We also assume a slotted time-axis, where a slot is exactly equal to the time to transmit one packet of either type. The presence of noise on the channel affects the transmission of the packets, so that they can be erroneously received. Packet errors can be revealed at the receiver's side by using, for example, a cyclic redundancy check (CRC) code [10]. This allows to send back a positive/negative acknowledgment of correct packet reception, denoted as ACK/NACK, respectively. Negative acknowledgments may trigger a selective retransmission, depending on their type. In a plain SR ARQ transmission of identical packets, this point would be obvious, but for a multimedia setup this point deserves further discussion, as we will argue later.

Packet errors induced by the channels are modeled according to a discrete-time Markov chain. For simplicity, we consider a two-state chain similar to what done in [4], [5], where states are denoted with 0 and 1 and represent error-free and erroneous channel, respectively. In other words, transmission when the channel chain is in state 0 is always successful, while in state 1 it always fails. The corresponding chain is depicted

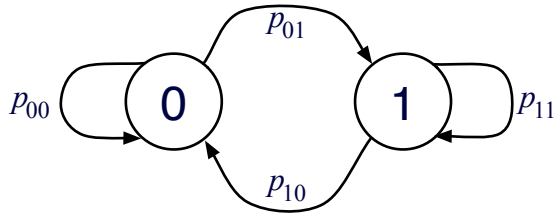

Fig. 1. The Markov representation of the channel (channel chain).

in Fig. 1 and can also be characterized through a transition matrix $\mathbf{P}$, collecting all transition probabilities p_{ij} from state $i \in \{0,1\}$ to $j \in \{0,1\}$, i.e.,

$$\mathbf{P} = \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix} . \tag{1}$$

The corresponding i-step transition probability can also be written as

$$\mathbf{P}(i) = \mathbf{P}^i = \begin{pmatrix} p_{00}(i) & p_{01}(i) \\ p_{10}(i) & p_{11}(i) \end{pmatrix} \tag{2}$$

Due to the fact that the p_{ij}s are probabilities, the matrix can be characterized by just two values; e.g., p_{10} is known once p_{11} is given, since $p_{10}+p_{11}=1$. Thus, instead of using the p_{ij}s, we will resort to a description through two alternative equivalent parameters, i.e., the steady-state packet error probability ε and the average error burst length B. These parameters can be derived as $\varepsilon = p_{01}/(p_{10}+p_{01})$ and $B = 1/p_{10}$, respectively.

We remark that, in spite of its apparent simplicity due to using just two states, the model is actually quite powerful since it enables the investigation of two different effects, i.e., average error rate (influenced by ε) and channel correlation (captured by B). Moreover, it would be rather straightforward to extend the model to a higher number of states, but it would also give more tedious computations and heavier notations; the main impacts of average error rate and channel correlation are already captured by the two-state model, though.

The time elapsed between the transmission of any packet and the reception of the feedback, either acknowledging the packet or not, is the *round-trip time* (rtt). By assuming that the system adopts a stringent time-out, we can guarantee that after one rtt the transmitter is always informed about the reception status of the packet. In this sense, the rtt is set equal to a fixed interval of time, which in turn corresponds to a given number of slots. In the following, this constant value, which is also sometimes referred as the ARQ window, is denoted by m. According to the discussion of [21], considering a variable rtt in the analysis would just lead to more complicated computations but basically achieve the same conclusions.

Other assumptions, which are quite common for ARQ studies, are as follows. We consider the network pipe to be in *Heavy Traffic* conditions [4], meaning the transmitter's queue is always full with packet ready to be transmitted, which is a reasonable assumption for a multimedia stream. We also consider the feedback messages (ACK/NACK) to be never in error. This is also a quite common assumption made by almost the entire totality of ARQ-related studies, motivated by their shorter size. Indeed, there are some investigations [16], [20] about the consequences of feedback errors, and their main conclusion is that the impact is marginal and mostly consists in a re-scaling of the channel error probability. That is, if the average direct error rate is ε and the average feedback error rate is ε_f, the performance of the system is roughly equivalent to that of a system with average error rate equal to $\varepsilon + \varepsilon_f$ and always correct feedback.

IV. The proposed Retransmission Mechanism and the Macroscopic Description

The basic idea of applying ARQ to video flows is that retransmissions cannot be blindly applied to all the packets. Since certain packets are more important than others, their retransmission should be prioritized [6]. Therefore, we propose the following ARQ approach, which is an extension of what proposed in [9], the extension being that we consider a significantly longer ARQ window, and therefore a higher number of retransmission opportunities.

To grasp the idea of the proposal, we resort to a numerical example, which is complex enough to give the details of the ARQ operations in a sufficiently descriptive manner, but is tractable enough to be solved without using high computational power. The general analysis would involve the manipulation of a Markov chain, and the subsequent inversion of its transition matrix, and therefore the complexity is heavily dependent on the number of states in the system. Such a value is exponential in the round-trip time m [4].

In the following, we refer to this scenario, which is general enough to be extended with similar computations to the cases of interest. We take the round-trip time m equal to 4 slots, and we take $F = 3$. This means that each A-packet is followed by two B-packets. At the beginning of the transmission process, the ARQ window of m slots is therefore occupied by one A-packet, its two B-packets, and the last slot is taken by another A-packet. Extending m to larger values would be easy, but we remark that, being $m > F$, we are able to keep track, at any instant, of the relationships across at least two different groups of packets.

As argued in [5], the ARQ evaluation requires to track the status of all the last m transmitted packets, also called *pending* packets, the reason being that their acknowledgment status is still unknown at the transmitter's side. We define a vector $\mathbf{b}$ of m binary (i.e., either 0 or 1) elements, which are equal to 0 if the corresponding packet is correct, 1 if it is erroneous. The value of b_m describes the packet presently in transmission, while the values of b_{m-j} describe the packets transmitted j slot ago. Thus, b_m is equal to the present channel state, while b_1 tells whether the transmitter is about to receive an ACK ($b_1 = 0$) or a NACK ($b_1 = 1$), since $m - 1$ slots ago the channel was in either good or bad conditions, respectively. It holds that the probability that $b_j = x$ and $b_{j+1} = y$, with $x, y \in \{0, 1\}$ is p_{xy} from (1).

Even though vector $\mathbf{b}$ contains the channel state, it does not offer a full characterization of the ARQ system evolution. Indeed, we also need to distinguish between a particular slot being used for the transmission of an A-packet or a B-packet. Moreover, we can have different system evolutions, and therefore different steady-state probabilities, depending on how A-packets and B-packets interact and are retransmitted. In the following, we make the following assumption about how SR ARQ is implemented in our system. Note that also this assumption can be made without losing generality, as it would be easy to extend the analysis to different SR ARQ implementations. However, the one we investigate is also related to what proposed in the literature by [6], [8], [9], and appears therefore to be sensible.

The proposal concerns to apply the SR ARQ rationale to A-packets only. This means that they are retransmitted, while B-packets that are in error are simply discarded after one attempt. There are several motivations to do so. First of all, it is evident that A-packets are more important, since their correct decoding is not only useful for them, but also for their related B-packets. Moreover, the introduction of too many retransmissions in the flow may lead to instability of the queue and/or unnecessarily long delays. Conversely, by retransmitting just the A-packets, not only do we limit retransmissions to a fraction of packets equal to $1/F$, but we are also able to design the system so that no additional delay increase is occurred, which is of key importance for multimedia real-time flows.

The further idea that avoids delay increases consists of letting retransmissions of A-packets to take the place of B-packets in the following group of packets, in a sliding-window fashion. In other words, whenever an A-packet needs to be retransmitted, the following group will discard a priori one B-packet to leave room for the retransmission of the previous A-packet. In this way, no additional delay is incurred, but the A-packets can only be transmitted up to F times.

To better understand the mechanism, refer to our example with m=4, F=3, and assume that the channel is continuously in error. The first group of packets, consisting of one A-packet and two B-packets will be entirely in error, but only the A-packet will be retransmitted. In the next group, one B-packet will be discarded to leave room for the retransmission of the previous A-packet, therefore the second group will consist of one (newer) A-packet, the old A-packet at its second retransmission, and finally a B-packet, associated with the newer A-packet. Assuming that also this group of packets is in error, the next group will consists of only A-packets, the first of which will be a new packet, then the retransmission of the A-packet already transmitted once, and finally the retransmission of the oldest A-packet that has already been transmitted twice.

As a result, the periodic structure of the multimedia content is not touched, since a new independent A-packet must be transmitted every m slots. Also, the position of an A-packet within the group will be representative of how many transmission attempts it experienced already. When an A-packet

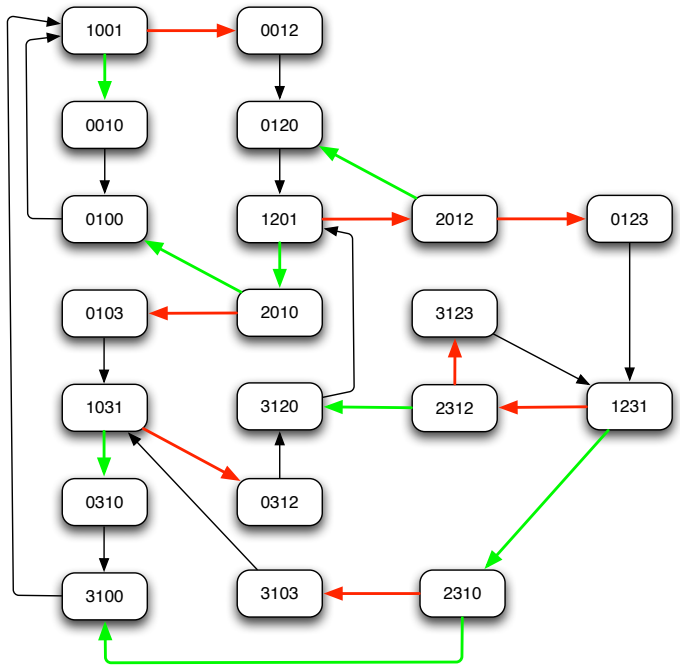

Fig. 2. Graphical display of the σ-transitions in the considered ARQ chain.

starting σ value	$\rightarrow \sigma'$ if b_1==0	$\rightarrow \sigma'$ if b_1==1
1001	0010	0012
0010	0100	0100
0100	1001	1001
0103	1031	1031
1031	0310	0312
0310	3100	3100
3100	1001	1001
0012	0120	0120
0120	1201	1201
1201	2010	2012
2010	0100	0103
3120	1201	1201
0312	3120	3120
3103	1031	1031
2012	0120	0123
3123	1231	1231
2312	3120	3123
2310	3100	3103
0123	1231	1231
1231	2310	2312

TABLE I
TABLE OF THE σ-TRANSITIONS

reaches the end of the group, it will be its last transmission attempt; if it still fails, it will be discarded.

To represent the evolution of this system, we utilize the concept of *system stage*, which means the configuration of A-packets and B-packets currently in the ARQ window, and their number-of-transmission indices, i.e., how many times they have been transmitted already. The stage is therefore denoted by an m-sized vector $\boldsymbol{\sigma}$, whose elements are integers. For notational simplicity, since we decided not to retransmit B-packets anyway, we set the jth element of $\boldsymbol{\sigma}$ as $\sigma_j = 0$ if the jth slot is occupied by a B-packet. If the packet is of type A, σ_j is equal to $1, 2, \ldots, F$ denoting the number of transmissions the packet already incurred (including the present one). We denote with Σ the total number of possible stages.

Thus, for our proposed example, the system starts at $\boldsymbol{\sigma} = [1001]$ since both A-packets (the one at the beginning of the ARQ window and the one at the end) are at their first transmission attempt. From there, the evolution of the stage actually depends on the channel. In particular, from $\boldsymbol{\sigma} = [1001]$, if $b_1 = 0$ the system will evolve to $[0010]$ (since the next transmitted packet will be of type B), and to $[0012]$ otherwise (since the next transmission will involve the retransmitted A-packet, at its second attempt).

The full evolution of the stage is called *macroscopic description* of the system [9], as it is a one-step evolution pattern that only indirectly depends on the channel evolution. Rather, it just requires to know whether the feedback packet that is about to be received will be an ACK or a NACK. In this sense, the same analysis would be possible also for more complicated systems, including Hybrid ARQ. For the system under investigation, the evolution of the macroscopic description is sketched in Fig. 2. To read the figure, note that green arrows represent situations where the transmitter receives an ACK, i.e., b_1=0 and the red arrows are instead cases where b_1=1 and a retransmission is scheduled. Whenever the evolution is identical in the two cases, which is what happens when $\sigma_1 = 0$, i.e., the oldest packet is of type B (so, it will not be retransmitted anyways, regardless of its status), the only exit transition is denoted by a black arrow. The full transition is also reported in a tabular form in Table I. For the system under consideration, $\Sigma = 20$.

Thanks to the Markov property of the channel, and the fact that also the macroscopic description $\boldsymbol{\sigma}$ evolves via one-step transitions, it is possible to frame the ARQ system into a Markov chain representation, with the state being $(\mathbf{b}, \boldsymbol{\sigma})$. The evolution of such a system from state $(\mathbf{b}, \boldsymbol{\sigma})$ to $(\mathbf{b}', \boldsymbol{\sigma}')$ can be promptly written as follows:

$$\text{Prob}[(\mathbf{b}, \boldsymbol{\sigma}) \rightarrow (\mathbf{b}, \boldsymbol{\sigma}')] = \begin{cases} p_{b_m b'_m} & \text{if } b_j = b'_{j-1} \text{for all } j \\ & \text{and } \boldsymbol{\sigma}' \text{ follows Tab. I} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The solution of this system can be found through matrix-geometric procedures [22]. We can collect all the transition probabilities in the transition matrix $\mathbf{T}$ of the Markov system. To this end, we can easily determine a single numerical label for each state, and therefore write an integer number j instead of $(\mathbf{b}, \boldsymbol{\sigma})$. In the following, we will write $b_k(j)$ to denote the kth element of the vector $\mathbf{b}$ in state j and similarly $\sigma_k(j)$ to refer to the kth element of its stage $\boldsymbol{\sigma}$. In general, the number of states, i.e., the range of index j or the size of $\mathbf{T}$, is $N = \Sigma \cdot 2^m$, which in our example would lead to 320 states.

The steady-state probabilities $\boldsymbol{\pi}$ of the system can be found by solving for $\boldsymbol{\pi}$ the system $\mathbf{T}\boldsymbol{\pi} = \boldsymbol{\pi}$ under condition $\sum \boldsymbol{\pi} = 1$. Thus, the vector of steady-state probabilities can be promptly derived as

$$\boldsymbol{\pi} = \begin{bmatrix} \mathbf{T} - \mathbf{I} \\ \mathbf{1} \end{bmatrix}^{-1} \mathbf{e}_{N+1} \,. \tag{4}$$

where $\mathbf{I}$ is an $N \times N$ identity matrix, $\mathbf{1}$ is an N-sized all-one

vector, and $\mathbf{e}_j$ is a vector with all zeros and a one in the jth position.

Since in our case $N = 320$, the resulting Markov chain and the corresponding transition matrix are easily manageable. More in general, even for larger N, the matrix $\mathbf{T}$ would be sparse, with only few non-zero elements. As visible from (3) there are indeed at most two non-zero elements per row out of N; thus, (4) is relatively simple to compute.

After finding the steady-state probabilities, several performance metrics can be evaluated. In this work, we focus on three classic performance metrics of ARQ systems, namely: throughput, average number of transmission attempts for a packet, probability of packet discarding. It is worth noting that the throughput considered in this paper is regarded as the amount of correctly delivered packets that can be reproduced at the receiver's side. Thus, we *do not count* a B-packet which is correctly delivered but it cannot be decoded since its corresponding A-packet is not received. In other words, we consider as throughput contributions only those A-packets that have been correctly delivered and the B-packets associated with them that have been correctly delivered as well. About retransmissions and packet discarding, we remark that for B-packets the computation is pretty simple, since the average number of transmissions would always be 1 and the average discarding rate is ε. Thus, we limit the analysis of those metrics to A-packets. Importantly, discarding a B-packet has only a minor effect on the video quality, while discarding an A-packet can freeze the multimedia flow for an entire frame duration.

The throughput, denoted as S, can also be seen as 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 a stage led by an A-packet, i.e., $\sigma_1 > 0$, we have three possibilities. If such a packet is immediately acknowledged, we increase the throughput by one packet; moreover, in this case we also count the amount of correct B-packets associated with it as a further throughput contribution. We do the same if the packet is not acknowledged, but the transition is toward another system state where it will be acknowledged. Conversely, if the packet is not acknowledged, and it is known that the next state will not acknowledge it either, we do not count (yet) any contribution. If the stage is led by a B-packet, we do not count any throughput contribution, as if the leading packet is in error we do not have to count it, and if it is correct it has already been counted when the A-packet associated with it was received. However, this just leaves out the case of an A-packet that is correctly received but only at after several unsuccessful transmission attempts. This would imply that all the past successful transmissions of B-packets associated with it need to be "recovered" and counted as throughput contributions. In our example, there is just one such case, which happens when the A-packet is correctly received only at its last transmission attempt (overall, there are 3 transmission attempts for an A-packet, and the first two are included in the reasoning above). If the first transmission of the A-packet takes place at time t, and it is erroneous, the second transmission (i.e., the first retransmission) will be described by the last position of σ at time $t+1$. Such a value is known by looking at the evolution of the Markov chain. The condition of the third transmission attempt is instead not included in the system state at time $t+1$, but it will happen m slots further in the future. Thus, we can exploit the memoryless property of Markov channel and compute its m-step transition to a good channel; in other words, we simply look at transition from 1 to 0 on $\mathbf{P}^m$. By looking at these values, we know the probability that the A-packet will be eventually correct; hence, we can also count as throughput contributions all the B-packets associated with an A-packet that is correctly received only at the last attempt.

We can formally write down this reasoning as

$$\begin{aligned} S \;=\; & \sum_{j=1}^{N} \Big\{ \chi\big(\sigma_1(j) > 0\big)\pi_j\big(1 - b_1(j)\big) \\ & + \chi\big(\sigma_1(j) == 1\big)\pi_j\big(1 - b_1(j)\big) \sum_{\ell=2}^{F} \big(1 - b_\ell(j)\big) \\ & + \chi\big(\sigma_m(j) == 2\big)\pi_j\big[\big(1 - b_m(j)\big) + b_m(j)p_{10}(m)\big] \\ & \cdot \sum_{\ell=m-F}^{m-2} \big(1 - b_\ell(j)\big) \Big\} \end{aligned} \tag{5}$$

where $\chi(\cdot)$ is an indicator function, equal to 1 if the argument is true, 0 otherwise. To better clarify the equation, the first row accounts for the contribution of leading A-packets, i.e., those for which $\sigma_1(j)$ is larger than 0. If the bitmap value $b_1(j)$ is 0, they are accounted as throughput contributions. The second row considers the contribution of B-packets following an A-packet which is correct at its first attempt; same as the previous row, but this time the contribution is counted not only if $b_1(j) = 0$ (this is still needed) but also whenever $b_\ell = 0$ too, where ℓ is one of the $F - 1$ indices following the first. The third row considers instead the case where an A-packet is the *last* of the window and it is at its second transmission attempt. Note that there is no need to condition on the transmission in the previous attempt being erroneous (i.e., the opposite case of the previous row): we already know it, from $\sigma_m(j)$ being 2. Again, the $F-1$ associated B-packets are counted if they are correct, but this time they are in positions from $m - F$ to $m - 2$, and in position $m - 1$ there is a newer A-packet, see Fig. 2. Moreover, the condition to count them is that either the A-packet itself is correct, i.e., $b_m(j) = 0$, or it will be m slots afterwards, which is the meaning of $b_m(j)p_{10}(m)$. Note that this computation stops at the third row only because we considered three transmission attempts. It would be straightforward, but much more cumbersome, to extend it to a higher retransmission limit, although that would also require to increase F as well.

The average number of transmissions experienced by an A-packet can be computed by identifying the steady-state probability α_j that the system is receiving feedback for an A-packet at its j transmission (not necessarily the last one), with $j \in \{1, 2, \ldots, F\}$, which is

$$\alpha_k = \sum_{j=1}^{N} \chi\big(\sigma_1(j) == k\big)\pi_j \; . \tag{6}$$

The α_j probabilities can be related to the probability q_j that an A-packet experiences exactly j transmissions before being either received or discarded. Such a relationship is a simple linear system of F equations in F unknowns. In our case, where $F = 3$, such a system reads

$$\begin{aligned} \alpha_1 &= \frac{q_1 + q_2 + q_3}{q_1 + 2q_2 + 3q_3} \\ \alpha_2 &= \frac{q_2 + q_3}{q_1 + 2q_2 + 3q_3} \\ \alpha_3 &= \frac{q_3}{q_1 + 2q_2 + 3q_3} \end{aligned} \tag{7}$$

which is easy both to solve and also to generalize to cases with a larger value of F. The average number of transmissions for an A-packet, denoted as N_a, can therefore be computed as

$$N_a = \sum_{j=1}^{F} j q_j \ . \tag{8}$$

Finally, the probability of discarding for an A-packet, denoted as $P_{dsc,A}$, is simply computed as the probability of finding an A-packet at its last transmission attempt, and noting that if the channel is bad it will be discarded. Thus,

$$P_{dsc,A} = \sum_{j=1}^{N} \chi\big(\sigma_1(j)==F \ \&\ b_1(j)==1\big)\pi_j \ . \tag{9}$$

V. Numerical results

We present now some numerical results that are directly derived from the analytical framework of the previous sections. We also run independent simulation tests (that is, running Monte Carlo evaluations of the Markov chains instead of solving them) to verify the analysis, and all the results were found to be in agreement. To avoid unnecessary overlap in the graphs, we will not show them, but the match is perfect as both analysis and the simulation refer to the same system and the analysis is exact.

Before discussing the individual result, we also remark that the numerical values of the performance metrics should be compared with a baseline scenario of SR ARQ applied to homogeneous packet flows. In such a case, it is well known from the theory [3] that the throughput is $1 - \varepsilon$, and also the frame discarding probability and the average number of packet transmissions can be computed relatively easily. In our case, we expect a lower value for the throughput, because of the interdependence of packet encoding. In other words, B-packets do not contribute to throughput both when they are erroneous, and in the case the related A-packet is erroneous. Thus, a fraction of packets equal to $(F-1)/F$, in our numerical example, two thirds, is unusable with a probability which is about double than the usual error rate (however, there is also a dependence on the channel correlation). Yet, as shown in the following, a simple mechanism that just retransmits the missing A-packet is able to compensate for this sort of error.

We present results for all the three metrics discussed in the previous section, plotting them versus the average error rate ε and the average error burst length B. In the latter case, we consider an "iid" case as the one for which $B = (1 - \varepsilon)^{-1}$,

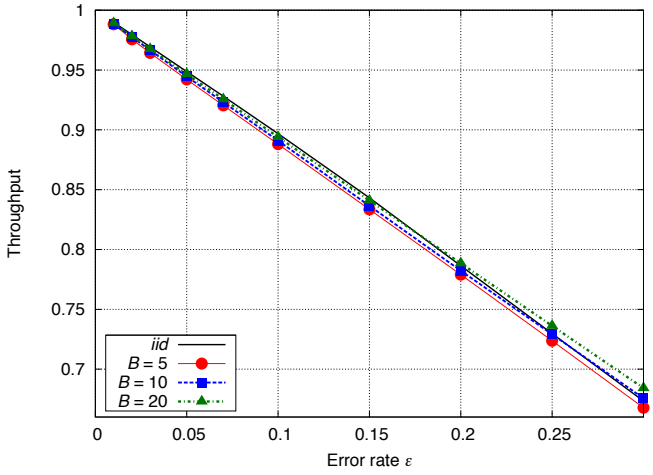

Fig. 3. Throughput of our proposed SR ARQ scheme, versus error rate ε.

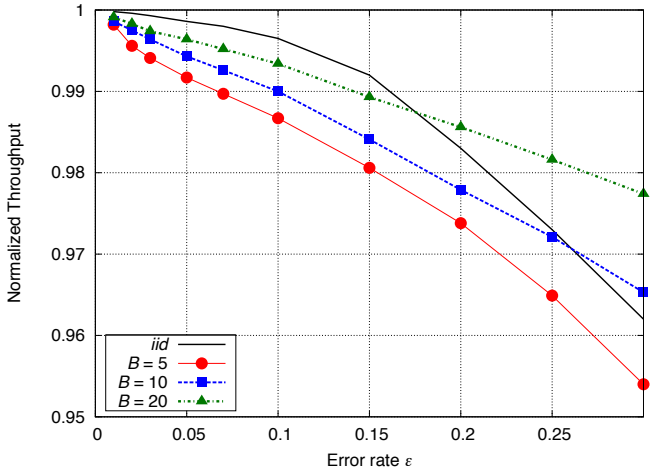

Fig. 4. Normalized throughput of our proposed SR ARQ scheme, versus error rate ε.

which is the average error burst length of an uncorrelated channel, i.e., with independent and identically distributed (i.i.d.) errors; in those cases, B is actually variable depending on ε. The resulting parametric analysis in ε and B serves to show the impact of errors in terms of both their frequency and their correlation, and enables discovering notable behaviors that can be useful to design multimedia transmission protocols.

Fig. 3 shows the impact of the average channel error rate on the throughput. As visible from the figure, the trend is almost linear and the curve is very close to $1 - \varepsilon$. This means that retransmission of just A-packets has been able to recover most of the throughput loss due to incremental encoding; in the absence of these retransmissions, the throughput would have been much lower. From the figure it is also possible to note that channel correlation affects the throughput in a non-trivial manner. In particular, at low error rates the throughput of a correlated channel is slightly lower than an iid channel. This is sensible, as the likelihood of recovering good B-packets whose reference A-packet was in error is lower; since the channel is correlated, then it is more likely that if the A-packet is wrong, the B-packets are wrong too.

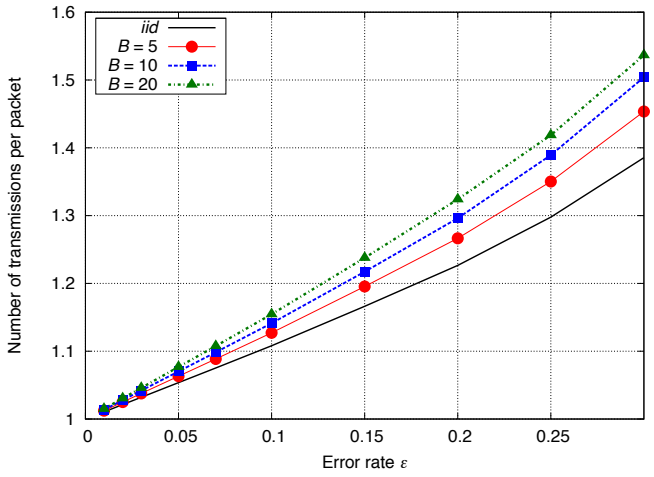

Fig. 5. Average number of transmissions of our proposed SR ARQ scheme, versus error rate ε.

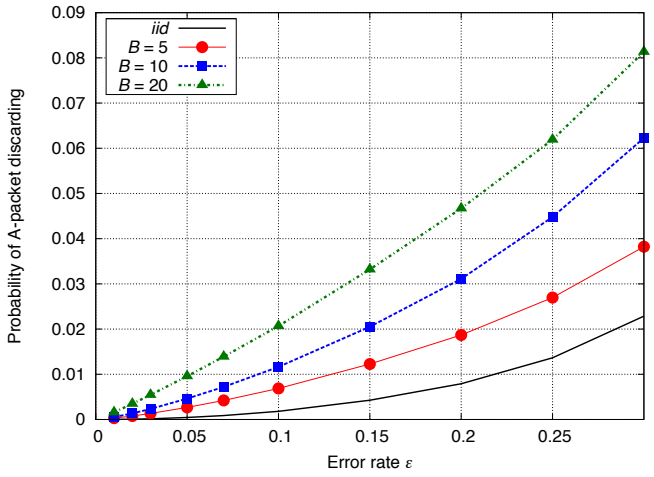

Fig. 6. Packet discarding probability of our proposed SR ARQ scheme, versus error rate ε.

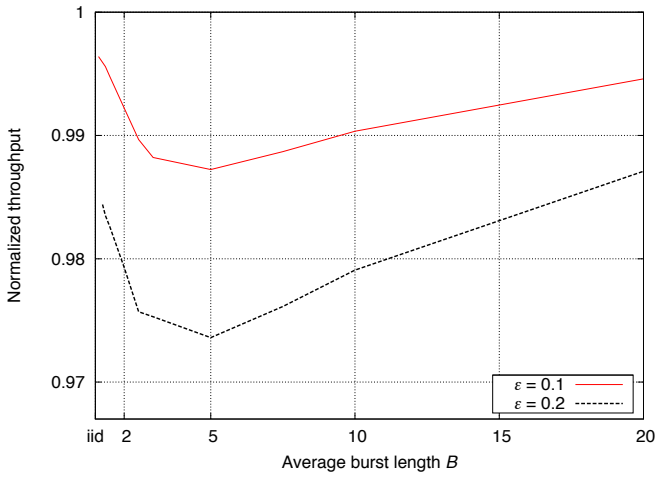

Fig. 7. Normalized throughput of our proposed SR ARQ scheme, versus average error burst length B.

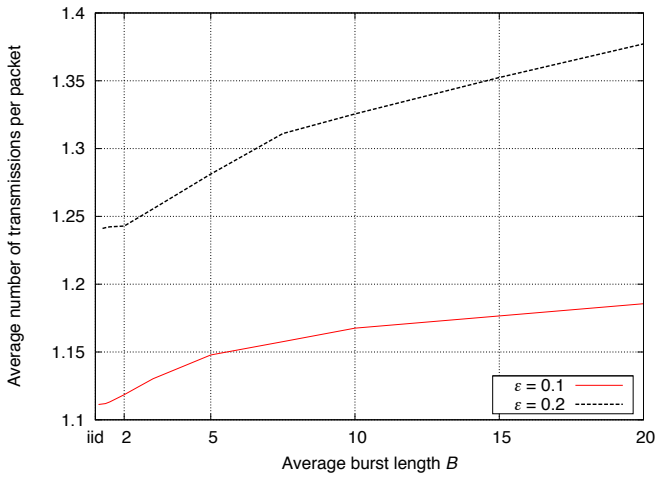

Fig. 8. Average number of transmissions of our proposed SR ARQ scheme, versus average error burst length B.

All the curves are very similar, though, therefore in Fig. 4 we plot a *normalized* throughput, where the normalization term is the reference value $1-\varepsilon$. It is visible that our proposed retransmission mechanism is able to lose only few percent with respect to the throughput of plain SR ARQ, at the price of only one retransmission over F packets. Also importantly, the worst scenario is a moderately bursty channel, $B = 5$, which is comparable with $m = 4$ and therefore in which sequences of errors often lead to losing an entire group of packets. Somehow surprisingly, higher correlation values lead to slightly higher throughput values (especially, the throughput is even higher than the iid case for high ε). The reason of this behavior is to be found in that a correlated channel stays in error for a longer time, but also has long sequences of error-free slots. Thus, a correlated channel may be a better scenario to transmit a multimedia flow, since it more often enables the correct reception of an entire group of packets.

Figs. 5 and 6 report instead the average number of transmissions for an A-packet, N_a, and the discarding probability of an A-packet, $P_{dsc,a}$. Note that N_a is the number of *transmission* so it starts from 1, as each packet gets at least transmitted once. The number of *retransmissions* would simply be N_a-1. These metrics exhibit a similar trend in that they become higher for increasing error rate and also for average error burst length.

From these results, it seems that throughput has a minimum when the burst length B is comparable with the rtt, and the probability of discarding and average number of retransmissions for an A-packet are increasing in B. To better investigate this trend, we also plot Figs. 7–9, where these values are plotted versus B, for two different values of ε. The left-most value on the x-axis, denoted as "iid," equals 1.111 and 1.25, when $\varepsilon = 0.1$ and $\varepsilon = 0.2$, respectively.

Fig. 7 reports the throughput value, again normalized to $1-\varepsilon$ as was for Fig. 4, and confirms that there is a minimum for the throughput around $B = 5$ (the closest to m of the considered values). Fig. 8 reports the average number of retransmissions. For the iid case, such a value is naturally $1/\varepsilon$. However, it increases when the correlation is stronger. A similar trend is exhibited by the probability

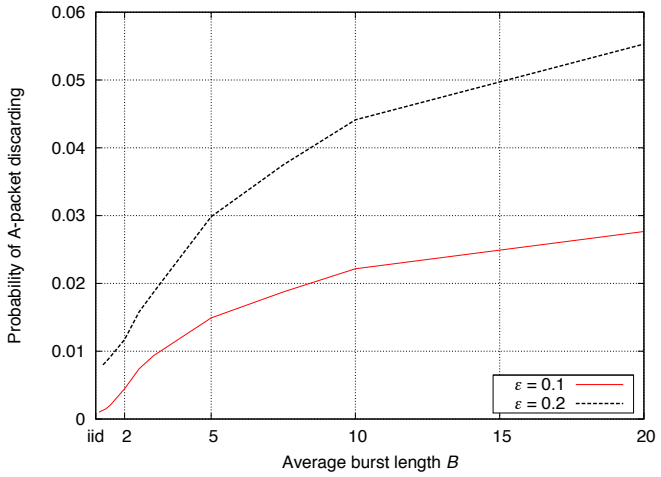

Fig. 9. Packet discarding probability of our proposed SR ARQ scheme, versus average error burst length B.

of discarding an A-packet, which is plotted in Fig. 9. Note that discarding an A-packet has the effect of losing an entire group of F packets, since also the B-packets cannot be played. Remarkably, while for moderate correlation the retransmission mechanism is able to keep this probability to low values with just 3 retransmissions (indeed, the discarding probability of an A-packet is obviously equal to ε^F), the value for stronger correlation is instead non-negligible.

VI. Conclusions

In this paper, we presented a detailed Markov analysis of the performance of SR ARQ for multimedia flows. The strong suit of our investigation is that we explicitly kept into account the relationships among the different packets determined by incremental encoding. This enabled us to show some non-trivial effect of applying ARQ to multimedia streams.

In particular, we gave the following important contributions. First, we identified a way to apply SR ARQ effectively to multimedia flows by being even more "selective" in the retransmission pattern, i.e., not only retransmitting erroneous packets only (as per the classical selective repeat ARQ), but also carefully choosing to retransmit only the independent packets and discarding the erroneous incremental packets. Moreover, through our analysis we were able to show that such an implementation of SR ARQ is effective in achieving good performance. Especially, we were able to show the dependence of the overall ARQ performance metrics on the average packet error rate and the channel correlation. Thus, these results can serve as practical guidelines and the approach can be extended to assess the performance of specific multimedia transmission protocols over correlated channels.

References

[1] D. Towsley and J. Wolf, "On the statistical analysis of queue lengths and waiting times for statistical multiplexers with ARQ retransmission schemes," *IEEE Trans. Commun.*, vol. 27, no. 4, pp. 693–702, 1979.

[2] A. G. Konheim, "A queueing analysis of two ARQ protocols," *IEEE Trans. Commun.*, vol. 28, no. 7, pp. 1004–1014, July 1980.

[3] S. Lin, D. J. Costello, and M. J. Miller, "Automatic-Repeat-reQuest error control schemes," *IEEE Commun. Mag.*, vol. 22, no. 12, pp. 517, 1984.

[4] 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.

[5] W. Luo, K. Balachandran, S. Nanda, and K. Chang, "Delay analysis of selective-repeat ARQ with applications to link adaptation in wireless packet data systems," *IEEE Trans. Wireless Commun.*, vol. 4, no. 3, pp. 1017–1029, May 2005.

[6] J.-B. Seo, Y.-S. Choi, S.-Q. Lee, N.-H. Park, and H.-W. Lee, "Performance analysis of a Type-II Hybrid-ARQ in a TDMA system with correlated arrival over a non-stationary channel," in *Proc. ISWCS*, 2005, pp. 59–63.

[7] 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.

[8] L. Badia, M. Levorato, and M. Zorzi, "Analysis of selective retransmission techniques for differentially encoded data," *Proc. IEEE ICC*, Dresden, Germany, 2009.

[9] L. Badia, N. Baldo, M. Levorato, and M. Zorzi, "A Markov framework for error control techniques based on selective retransmission in video transmission over wireless channels," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 3, pp. 488–500, Apr. 2010.

[10] 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.

[11] A. Larmo, M. Lindström, M. Meyer, G. Pelletier, J. Torsner, H. Wiemann, "The LTE link-layer design," *IEEE Comm. Mag.*, vol. 47, no. 4, pp. 52–59, Apr. 2009.

[12] 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.

[13] 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.

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

[15] K. Stuhlmüller, N. Farber, 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.

[16] R. Cam and C. Leung, "Throughput analysis of some ARQ protocols in the presence of feedback errors," *IEEE Trans. Commun.*, vol. 45, no. 1, pp. 35–44, Jan. 1997.

[17] J. G. Kim and M. M. Krunz, "Delay analysis of selective repeat ARQ for a Markovian source over a wireless channel," *IEEE Trans. Veh. Technol.*, vol. 49, no. 5, pp. 1968–1981, 2000.

[18] 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.

[19] 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.

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

[21] ——, "A Markov analysis of selective repeat ARQ with variable round trip time," *IEEE Comm. Lett.*, vol. 17, no. 11, pp. 2184–2187, Nov. 2013.

[22] M. F. Neuts, *Matrix-Geometric Solutions in Stochastic Models*. New York: Dover Publications, inc., 1981.