On the delay statistics of an aggregate of SR-ARQ packets over Markov channels with finite round-trip delay

M. Rossi, L. Badia, M. Zorzi
{mrossi,lbadia,mzorzi}@ing.unife.it
University of Ferrara, via Saragat 1, I-44100 Ferrara, Italy

Abstract— **In this paper we investigate the delay statistics of an aggregate of fully reliable Selective-Repeat ARQ packets. The sender transmits packets whose error process is characterized by means of a two-state Discrete Time Markov Channel (DTMC). At the receiver these packets are checked for errors and ACK/NACK messages are sent back to the sender accordingly. No errors are accounted for in the reverse channel. The feedback message is assumed to be known at the transmitter m channel slots (round-trip delay) after the packet transmission started. An appropriate Markov model has been previously developed in order to find the exact statistics of the delays experienced by ARQ packets. This work presents an extension of the analysis that computes the delay statistics of an aggregate of ARQ packets. This is achieved without increasing the model complexity and allows useful considerations from the point-of-view of higher level protocols.**

I. Introduction

When data are transmitted over channels with high error probabilities, mechanisms are required to recover from errors, which may occur more frequently than the application can tolerate. In usual protocol stacks, error control is performed at multiple levels, e.g., at the physical layer by error correction codes, at the data-link layer by ARQ techniques, as well as at the transport layer by TCP. In order to cut the right tradeoff between data reliability, latency, and efficient bandwidth usage, error control techniques must be carefully designed and their performance well understood.

The study of ARQ error control techniques in wireless systems has not enjoyed great popularity in recent years compared to, e.g., error correction coding strategies. This is mainly due to the type of application envisioned in these systems, i.e., voice and circuit-switched data, where strict delay guarantees are provided. With the extension of packet data and Internet services over wireless links, the increased delay tolerance of many applications and protocols leads to a paradigm shift, where error recovery by retransmission may be more efficient than protecting all data a priori by means of costly FEC. Note that one of the features of a well-designed wireless communications system is that the channel quality is generally good with occasional extremely severe drops, i.e., the best scenario for retransmission protocols.

The key point, when ARQ solutions are considered is that they directly interact with higher level, by determining both delay/jitter performance and error probability of higher level packets. For these reasons, their correct configuration is key in achieving the needed higher level QoS. In the TCP/IP specific case, for example, the crucial point is to keep the IP packet delivery time under a certain threshold level, after which the TCP enters timeout without being able to activate the fast recovery phase (this, in some cases would allow a faster recovery from lost packets by avoiding wasting of resources, i.e., without drastically decreasing the congestion window). Moreover, in the case of real-time data traffic (e.g. video-conference, multicast sessions), at the receiver playout buffer late incoming packets are discarded, thereby leading to the waste of the bandwidth and power needed for their transmission. Hence, an accurate study of the delivery delay process at the ARQ level appears to be a crucial point in order to understand the interaction between the higher level performance and the link layer retransmission process.

In ARQ, the transmitter sends packets (PDUs) consisting of payload and error detection codes. At the receiver side, based on the outcome of the error detection procedure, acknowledgment messages are sent back to the transmitter (ACK or NACK, according to the result of error detection). The sender performs packet retransmissions based on such acknowledgments. In general, ARQ protocols are variants of the following basic schemes: stop-and-wait (SW), go-back-N (GBN) and selective repeat (SR). In SW, only one packet in a round-trip delay time is transmitted, i.e., a new packet is transmitted only when the ACK of the current one is received. This scheme is not very efficient, especially when the round-trip delay is large. In GBN, packets are transmitted continuously, without waiting for acknowledgment messages. When a NACK is received, the transmitter simply retransmits the erroneous packet and all the subsequent ones. The SR scheme, that is considered in this work, is the most efficient. Here packets are transmitted continuously, but only negatively acknowledged packets are retransmitted, i.e., retransmissions are selectively triggered by NACK messages. We observe that, when the round-trip delay goes to zero all the presented schemes become identical. According to [3] and [6] we refer to this situation as *ideal SR* ARQ scheme.

In the presence of the ARQ protocol, we can subdivide the overall PDU delay in three contributions. The first is due to the queueing delay in the source buffer, i.e., the time between the PDU release by higher levels and the instant of its first transmission over the channel. This term depends on both the channel behavior and the PDU arrival process. The second contribution

This work has been partially supported by ERICSSON Research.

is the time between the first transmission and the correct reception of the PDU. This term only depends on the channel behavior. The last delay is due to the time spent in the receiver re-sequencing buffer. In fact, even if the sender transmits packets in order, due to random errors and consequent retransmissions, they can arrive out of sequence. Hence, correctly received PDU with higher identifier must wait in the receiver re-sequencing buffer until all the PDUs with lower identifier have been correctly received. This last term is the most complicated because, by considering a tagged PDU, it depends on errors experienced by all PDUs sent in the same round-trip in which the tagged one has been transmitted for the first time. These quantities will be referred to as *queueing delay*, *transmission delay* and *re-sequencing delay*, as usually done in the literature [6]. In addition, we define *delivery delay* as the sum of the second and third term.

Several studies have been performed on the delay performance of the SR protocol over a wireless channel [3][6][7][9][11][12][13]. In [7] Konheim derived the exact distribution of PDU delays with finite round-trip delay, but considering an *i.i.d.* error process. In [9] Anagnostou and Protonotarios who also consider a static channel, propose an alternative approach considering a Bernoulli arrival process. Rosberg and Shacham [11] analyzed the re-sequencing delay and buffer occupancy at the re-sequencing buffer considering an *i.i.d.* channel. Rosberg and Sidi in [12] analyzed the joint distribution of transmitter and receiver buffer occupancies over a static channel. In [13] Zorzi and Rao, also considered the *ideal SR* scheme, here the effectiveness of Markov model has been proved by means of comparison with a simulated fading process. The time varying channel has been investigated for the first time by Fantacci in [3], where the *ideal SR* scheme is considered. This approach leads to an analytical lower bound on delay performance with respect to the situation of a finite round-trip delay. In [6] Kim and Krunz accounted for a time varying channel, a finite round-trip delay and a Markovian traffic source. Here, a mean analysis is developed for all the ARQ delay contributions, in the computation of the source queueing delay the *ideal SR* is considered and the mean re-sequencing delay is obtained from an approximate approach and by considering *Heavy traffic* condition.

In this paper, we study the delay performance of a fully reliable SR ARQ scheme, in the presence of a time varying channel [1][2][5] and of a finite round-trip time. A *Heavy traffic* packet source and reliable ACK and NACK messages are assumed. In [8] an exact analysis for the computation of the delivery delay statistics of SR-ARQ packets is presented, which accounts for a finite round trip delay and for a time varying channel. Here, we present an extended analytical approach, that is developed in order to find the *delivery* delay statistics of an aggregate of K packets.

The remaining part of the paper is organized as follows: in Section II the ARQ policy and the channel model are described. In Section III we present the computation of the delivery delay statistics to the case of an aggregate of ARQ packets. In Section IV results are reported and finally, in Section V, some conclusions are given.

II. MODEL FOR ARQ QUEUEING AND TRANSMISSION PROCESSES

We consider a pair of nodes, say a transmitter and a receiver, that communicate data packets through a noisy wireless link and use a fully reliable Link Layer protocol (unlimited retransmission attempts) to counteract channel impairments. In the forward direction data packets (ARQ PDUs) flow, while in the backward one only ACKs and NACKs flow (ACKs and NACKs are assumed error free). Moreover, we assume that both transmitter and receiver have unlimited buffer size and they adopt the following Selective-Repeat ARQ protocol (a generalization of the protocol described in [4]) at the Link Layer.

The sender continuously transmits new PDUs in increasing numerical order as long as ACKs are received. After each PDU reception, the receiver checks for packet errors and replies with an ACK/NACK accordingly. When the generic PDU is transmitted, say PDU i, the sender must await an ACK message for that packet until after it finishes the transmission of up to $m-1$ subsequent PDUs (new or retransmitted), $1 \leq m \leq \infty$, where m is the round-trip delay (m is commonly referred to in the literature [10] as the *ARQ window size*). At this point, PDU i must be retransmitted (in the case of NACK) or a new PDU has to be sent (in the case of ACK). The time is slotted and the slot time corresponds to a single PDU transmission. The wireless channel is characterized by means of a two-state Discrete Time Markov Chain (DTMC). Let 0 and 1 denote successful and erroneous PDU transmission in a given slot, respectively, and let:

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

be the channel transition probability matrix. The steady-state channel error probability is given by:

$$\varepsilon = \frac{p_{01}}{p_{10} + p_{01}} \tag{2}$$

while the average error burst length is given by $b = \frac{1}{p_{10}}$. We define as $\mathbf{P}(i)$ the i-step transition probability matrix, computed as follow:

$$\mathbf{P}(i) = \mathbf{P}^i = \begin{pmatrix} p_{00}(i) & p_{01}(i) \\ p_{10}(i) & p_{11}(i) \end{pmatrix} \tag{3}$$

Moreover, we consider a *Heavy Traffic* condition, i.e., once a PDU is correctly transmitted, a new one is always present in the source buffer. This assumption is justified by taking into account a TCP file transfer (FTP-like session or video/audio continuous data streaming) as packet source at the transmitter. The reliable ARQ completely avoids TCP timeouts (when the channel error is not too large) and the TCP level, after filling the bandwidth-delay product, behaves as a continuous packet source (the TCP window size is not decreasing because error recovery is never triggered).

As a last observation, where the *Heavy Traffic* assumption is not verified, the delivery delay computed with our model is an

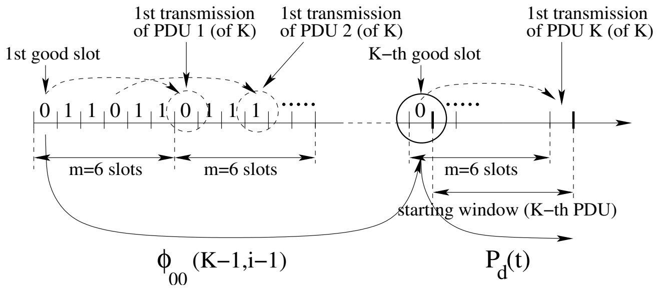

Fig. 1. Transmission of K ARQ PDUs over the channel.

analytical upper bound. Hence, our approach can be viewed as a worst case analysis, that becomes exact whenever the source packet generation rate is greater or equal than the PDU delivery rate (depending on the channel error process).

III. Computation of the delivery delay statistics of an aggregate of ARQ PDUs

In [8] an exact analysis of the delivery delay statistics of the SR-ARQ scheme has been presented. Figure 2 shows such statistics $P_d[t]$ for different values of ε in a correlated case ($b = 3$). In this Section we use the delivery delay statistics $P_d[t]$ described in this work, in order to obtain the delivery delay statistics of an aggregate of K ARQ PDUs: as will be shown in the following K is not a limiting factor for the model complexity.

In any event, our task here is to find out the delivery delay statistics of such an aggregate of K packets. With the term *delivery delay statistics*, we mean the number of slots elapsed between the time in which the first of the K PDUs is transmitted for the first time over the channel and the slot where all the K packets are correctly received and none of them is waiting for *in-order* delivery, i.e., all the K PDUs can be passed *in order* to higher levels.

The key point of the analysis is that the transmission of the first PDU (of K) in a given window position implies a correct transmission m slots earlier (in the same window position, see Fig. 1). Moreover, when a PDU is lost in a given slot, that PDU will surely be retransmitted m slots later in the same window position. So, each PDU is transmitted/retransmitted in the same window position until success, and only from this point on, the window position occupied by that PDU can be used to transmit a new packet. Before we proceed, let us introduce $\phi_{ij}(k,n)$, $i,j \in \{0,1\}$, that is the probability that there are k successful slots in $\{0,1,\ldots,n-1\}$ and that the channel state is j at time n, given that the channel state was i at time 0. $\phi_{ij}(k,n)$ can be computed recursively as follows (see [14]):

$$\begin{aligned}\phi_{ij}(k,n) &= \phi_{i0}(k-1,n-1)p_{0j} + \phi_{i1}(k,n-1)p_{1j} + \\ &+ \delta_{i,j}\delta(k)\delta(n) \end{aligned} \tag{4}$$

where $\phi_{ij}(k,n) = 0$ for negative values of either k or n, $\delta_{ij} = 1$ if $i = j$ and zero otherwise, and $\delta(k) = \delta_{k,0}$. A closed form for that function can be found in [15].

The key point is the following: each correct transmission in a given round enables a new transmission in the same window position in the following round[1], so counting the number of slots in which the channel is error-free in a given time interval, say $[i, i+N]$, is equivalent to counting how many new packets are transmitted over the channel in the interval[2] $[i+m, i+N+m]$. In practice the process given by new transmissions is simply the *enabling process* deterministically shifted forward in time by m slots (see Fig. 1). Hence, the probability distribution regarding the transmission of K packets is the same characterizing the enabling of K new packet transmissions. This last distribution, i.e., the probability distribution of the time i elapsed between the slot in which the first transmission of the first PDU (of K) is enabled and the instant in which the transmission of the K-th packet is enabled is given by $\phi_{00}(K-1, i-1)$, that is (the probability of having K good slots over i, $i \geq 0$, see Eq. (4)).

In the sequel we refer to this probability as $P_{en}[K,i]$. $P_{en}[K,i]$ can be computed as follows:

$$P_{en}[K,i] = \begin{cases} 0 & i < K \\ \phi_{00}(K-1, i-1) & i \geq K \end{cases} \tag{5}$$

Note that here, PDU K plays the role of the tagged PDU in the single PDU delivery delay evaluation, so the slots in which PDU K is enabled and transmitted can be viewed as slot 0 and m, respectively. Furthermore, all the out-of-order packets that PDU K has to eventually wait for after its correct transmission are only the ones transmitted between slot 1 and slot m (Fig. 1), i.e., the memory needed to account for the in-order delivery of all the K PDUs does not depend on K, but only on the value of the round trip delay (m).

In conclusion, we can subdivide the delivery delay of the K PDUs in two parts: in the first part we account for the time between the enabling of the first transmission of PDU 1 and the slot in which the transmission of PDU K is enabled ($\phi_{00}(K-1, i-1)$). In the second part, we track the delivery delay regarding the last (K-th) PDU, starting from the slot where the transmission of such PDU is enabled. Moreover, remember that *enabling process* and *new transmission process* are the same process deterministically shifted in time by m slots, and that the delivery delay regarding PDU K is characterized by the function $P_d[t]$ that starts to track the channel m slots before the first transmission of that packet, i.e., in the last slot tracked by $P_{en}[K,i]$. Thus, the distribution characterizing the delay between the instant of the first transmission of PDU 1 and the instant in which all K PDUs can be correctly delivered in-order can be found as:

$$P_d[K,t] = \begin{cases} 0 & s < K \\ \sum_{i=0}^{t} P_{en}[K,i] P_d[t-i] & \text{otherwise} \end{cases} \tag{6}$$

where $P_d[t]$ is the delivery delay statistics regarding a single PDU (see Figure 2).

[1] Hereinafter, we refer to this process as new transmissions *enabling process*.

[2] Because each error-free slot enables the transmission of a new packet m slots apart.

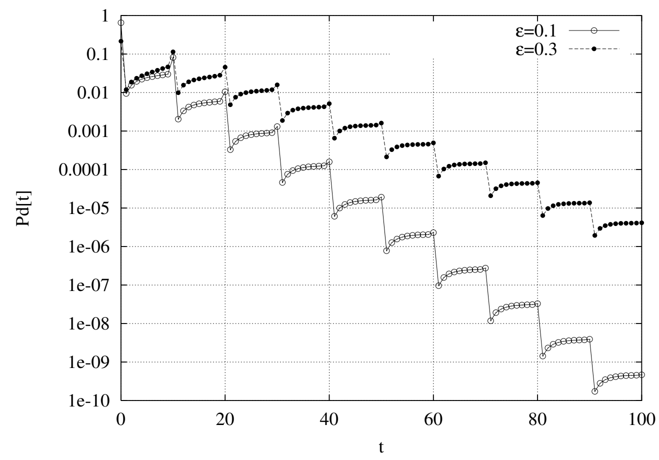

Fig. 2. Single PDU delivery delay distribution, $m = 10$, $b = 3$.

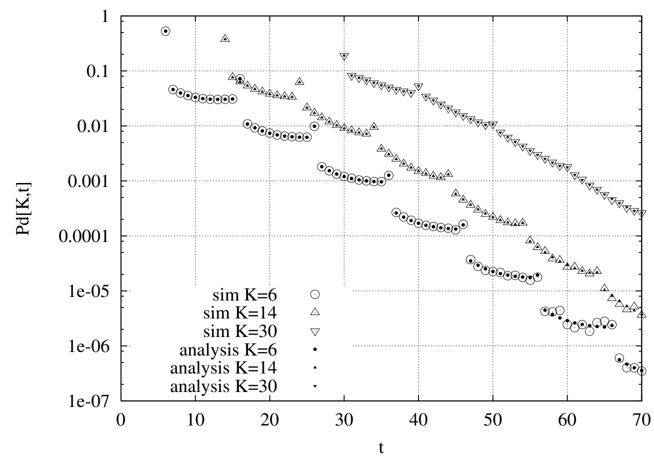

Fig. 3. Aggregate delivery delay distribution, $m = 10$, $\varepsilon = 0.1$. Comparison between analysis and simulation with $b = 3$.

IV. RESULTS

In order to better clarify the results presented in this Section, let us remember that what we call $P_d[t]$ is the probability to resolve the initial window (the one comprising the first transmission of the tagged packet and the $m-1$ previous transmissions) in a given number of slots ($k \geq 0$). The tagged packet delivery delay, instead, is the number of slots elapsed between the first transmission of the tagged packet and the instant in which it is released by the receiver re-sequencing buffer. The two delays above only differ for the sum of path delay and physical layer processing delay that are both constant terms. In the following, we will refer to $P_d[t]$ as the tagged packet delivery delay statistics by keeping in mind this difference.

In Figure 2 the delivery delay statistics $P_d[t]$ regarding a single PDU has been reported. It has been computed according to the analysis in [8], and it is used in the next as basis to compute the correspondent statistics of an aggregate of K PDUs. From Figure 2 it is interesting to note the cyclic behavior of the curves, which present a peak at the end of each round, i.e., when t is an integer multiple of m.

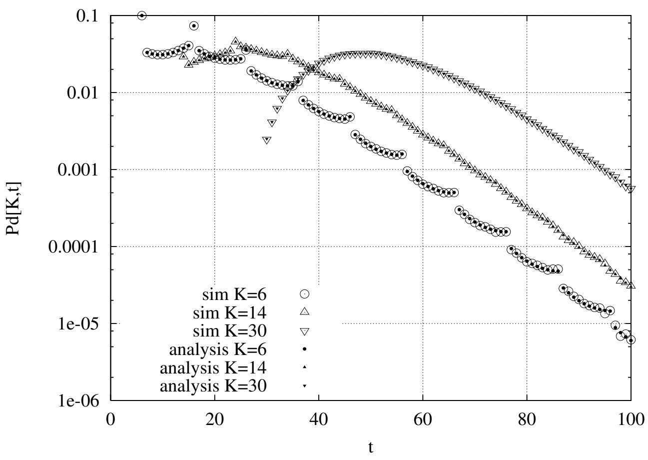

Fig. 4. Aggregate delivery delay distribution, $m = 10$, $\varepsilon = 0.3$. Comparison between analysis and simulation with $b = 3$.

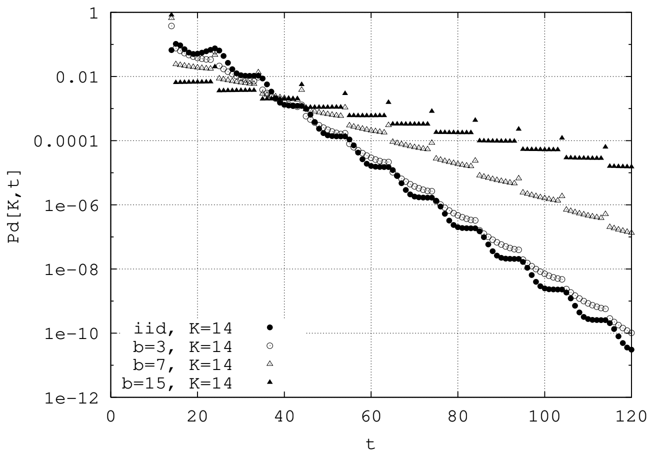

Fig. 5. Aggregate delivery delay distribution, $m = 10$, $\varepsilon = 0.1$. Comparison between different values of the channel correlation.

In Fig. 3-4 we report the delivery delay distribution of an aggregate of K PDUs by varying K, with two different values of the error probability ε. Simulation points are reported to show the correctness of the analysis[3]. Note that the cyclic behavior, that is characteristic of the single PDU delivery delay, in the aggregate tends to vanish: in fact, in this last case, there is a linerization effect due to the discrete-time convolution in Eq. (6). This phenomenon is more evident when both the aggregate size (K) and ε are to a high value.

In Fig. 5 we report $P_d[K,t]$ with $K = 14$ by varying the channel correlation. We can see that the burstiness of the channel heavily affects the performance: in fact, even when the error probability is kept constant, the behavior of $P_d[K,t]$ is very different, depending on b. In particular, note that the curve for *iid* channel does not represent a good approximation, especially in the cases with high b. Moreover, the point $(K, P_d[K,K])$ (the leftmost of each curve) has a different placement in the *iid* case

[3]Simulation fails at low ($P_d[t] \leq 10^{-5}$) probabilities due to the rare occurrence of the corresponding events and to the finite simulation time.

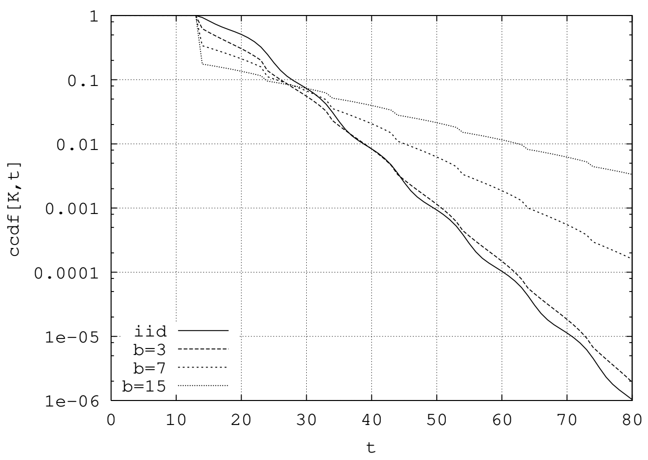

Fig. 6. Cumulative complementary delivery delay distribution, $ccdf[K, t]$ ($\varepsilon = 0.1, m = 10, K = 14$).

with respect to every correlated cases, even with low b. Thus, many values, e.g., the probability of resolving the entire aggregate exactly in K transmission are heavily underestimated if a correlated channel is erroneously assumed to be independent. On the other hand, a model, in which the burstiness of the channel is considered, is needed, or the system will be conservatively estimated.

By comparing Figure 2 and Figures 3–5, one may observe similarity in shape and behavior between $P_d[K, t]$ and $P_d[t]$. However, there are differences: $P_d[K, t]$ is right-shifted by $K - 1$ slots with respect to $P_d[t]$ and it presents a smoother behavior than the one characterizing $P_d[t]$. The shift is due to the minimum time needed to transmit all the K PDUs over the channel (K slots), whereas the smoothness is a property of the discrete convolution product (Eq. (6)).

We can derive also the cumulative complementary distribution of the delivery delay statistics for the aggregate case, $ccdf[K, t]$. This is the probability that the delivery delay exceeds x slots, formally:

$$ccdf[K, t] = 1 - \sum_{x=0}^{t} P_d[K, x] \quad (7)$$

This distribution is reported in Fig. 6 for $K = 14$, $m = 10$, $\varepsilon = 0.1$ by varying the channel burstiness b.

The function $ccdf[K, t]$ is very important because it is directly related to the delivery time of higher level packets. In the TCP/IP case, for example, it could be used to compute the timeout event probability, i.e., the probability that a given higher level packet experiences a delay larger than the value of the TCP timeout. Thus, the performance of the TCP protocol is directly related to this probability. Hence, the QoS perceived by the final user is directly related to this metric as well.

In addition, in the UDP case the cumulative distribution could be useful to decide whenever the delay perceived by higher level packets fulfills application requirements (e.g. in the case of voice traffic we have some delay constraints due by the maximum tolerable interactive delay). In any case, this statistics can be easily translated into performance metrics for any kind of transport protocol/application operating at higher levels.

Another challenging point is the connection between the function $ccdf[K, t]$ and the buffer size, since the buffers have to be dimensioned in order to avoid overflow events. Thus, given a maximum tolerable overflow probability, this could be directly related to the curves presented in Figure 6 to find the desired buffer size.

Note that both the aggregate statistics and its cumulative distribution as K increases could be well described by very simple heuristics (straight lines). Moreover, in 3G communication system, such as UMTS, a common value for IP packet segmentation is $K \approx 14$ (the value considered in Fig. 6). Hence, in this case, the use of very fast and low-complexity delay statistics estimates appears to be possible. Finding such heuristics can be an interesting topic for future research.

V. Conclusions and future work

In this paper we studied the delivery delay performance of a Selective Repeat ARQ scheme over a two-state Discrete Time Markov Chain. We derived the analytical expression of delivery delay distribution regarding an aggregate of ARQ packets as may occur, e.g., in IP packet segmentation. In the final part of the paper the distributions and their main characteristics are compared for several values of the channel error probability and error correlation. Moreover, their role in the evaluation of higher level protocol performance is discussed.

Further developements of this work can be investigation for heuristics that approximate the exact behavior, direct performance analysis of protocols at higher levels, e.g. TCP/IP performance, or studies regarding the correct dimensioning of link layer buffers.

References

[1] M. Zorzi, R.R. Rao, L.B. Milstein, “Error statistics in data transmission over fading channels,” *IEEE Trans. Comm.*, vol. 46, pp. 1468-77, November 1998.

[2] M. Zorzi, R.R. Rao, L.B. Milstein, “On the accuracy of a first-order Markov model for data block transmission on fading channels,” in Proc. IEEE ICUPC’95, pp. 211-215, November 1995.

[3] R. Fantacci, “Queueing analysis of the selective repeat automatic repeat request protocol for wireless packet networks,” IEEE Trans. Veh. Technol., vol. 45, pp. 258-264, May 1996.

[4] S. Lin, D.J. Costello, M.J. Miller, “Automatic-repeat-request error control schemes,” IEEE Commun. Mag., vol. 22, n. 12, pp. 5-17, December 1984.

[5] M. Zorzi, R.R. Rao, “On the Statistics of Block Errors in Bursty Channels,” *IEEE Trans. Commun.*, vol. 45, pp. 660-667, June 1997.

[6] J. G. Kim, M. M. Krunz, “Delay analysis of Selective Repeat ARQ for a Markovian Source Over a Wireless Channel”, IEEE Trans. Veh. Tech., vol. 49, pp. 1968-1981, Sept. 2000

[7] A. G. Konheim, “A queueing analysis of two ARQ protocols,” IEEE Trans. Commun. vol. 28, pp. 1004-1014, July 1980.

[8] M. Rossi, L. Badia, M. Zorzi, “Exact delay statistics of ARQ packet delivery delay over Markov channels with finite round-trip delay,” submitted for publication to IEEE ICC 2003.

[9] 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, February 1986.

[10] D. Bertsekas, R. Gallager, “Data Network,” Prentice Hall, 1992.

[11] Z. Rosberg and N. Shacham, "Resequencing delay and buffer occupancy under the Selective Repeat ARQ," IEEE Trans. Inform. Theory, vol. 35, pp. 166-173, January 1989.

[12] Z. Rosberg, M. Sidi, "Selective-Repeat ARQ: The joint distribution of the transmitter and the receiver resequencing buffer occupancies," IEEE Trans. Commun., vol. 38, pp. 1430-1438, September 1990.

[13] M. Zorzi, R.R. Rao, "On channel modeling for delay analysis of packet communications over wireless links," 36th Annual Allerton Conference, Allerton House, Monticello, IL, September 1998.

[14] R. A. Howard, "Dynamic Probabilistic Systems," New York: Wiley, 1971.

[15] M. Zorzi, R.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, October 1999.