

# Improving TCP over Wireless through Adaptive Link Layer Setting

C.F. Chiasserini, M. Meo

Dipartimento di Elettronica, Politecnico di Torino  
 C.so Duca degli Abruzzi 24, 10129 Torino - Italy  
 Tel.: +39-011-5644183, Fax: +39-011-5644099  
 Email: {chiasserini,michela}@polito.it

**Abstract**—Consider a communication link where the last hop is wireless and TCP is used as transport protocol over the end-to-end connection. We study the capability of the link layer to hide losses over the wireless link to TCP in spite of the time varying transmission quality. We focus on link-layer retransmission mechanisms and determine their parameter setting in such a way that a reliable communication link is provided. In particular, chosen a significant QoS metric at the transport layer and fixed its targeted value, we adapt the maximum number of link-layer transmissions to the characteristics of the wireless link so that the desired QoS at the transport layer is provided. Results showing the impact of the link-layer retransmissions on TCP performance are derived by using analytical models based on Markovian techniques.

## I. INTRODUCTION

One of the most challenging aspects in telecommunication networks is to provide users with the wireless access to Internet. Indeed, as end systems connect through a radio link, users can access Internet-based applications while freely moving over the network area; however, the transmission quality over the radio channel varies considerably over time and may significantly affect the performance of protocols supporting reliable data transport, such as TCP. Consider a network scenario where the last hop of a communication path is wireless. Whenever periods of heavy noise take place, TCP interprets data losses as signs of network congestion and initiates its retransmission policy, which may seriously degrade throughput [1]. To avoid such performance degradation, link layer solutions can be used to obtain local data reliability and make the wireless link appear to TCP as a more reliable link, although with a longer and variable delay [1], [2], [3], [4], [5], [6], [7], [8].

In this paper, we focus on the capability of link layer error-recovery functions to hide losses over the wireless link to TCP by adapting their parameter setting to the radio channel conditions. This approach provides a reliable wireless link in spite of the heterogeneous environments mobile terminals may incur, and it enables an efficient use of TCP over wireless connections without any modification to the transport protocol.

A graphical representation of the system under study is given in Fig. 1. We focus on a single TCP connection where the receiver resides at a mobile terminal. Before reaching the fixed network access point, hereinafter called Base Station (BS), TCP segments traverse the wired network from which they perceive an average delay  $D$  and average loss probability  $L$ . At the BS they rely on the Link Layer (LL) protocols to get to the TCP receiver through the wireless link. The system performance is evaluated through Markovian models of the protocols at the link layer and of TCP NewReno at the transport layer. The LL and

the TCP models are solved by means of a fixed-point iterative procedure [9], [10], as sketched in the lower part of Fig. 1 (see Sec. II-B for further details).

Selected a significant QoS (Quality of Service) metric and fixed the corresponding targeted value, our analytical framework provides the LL parameter setting that fulfills the QoS requirements as the characteristics of the fading process change. We take the loss probability of the TCP data segments as target QoS index and derive the value of maximum number of transmissions at the link layer that provides the desired QoS as the error probability and the correlation of the fading process over the wireless link vary. The effect of varying the maximum number of transmissions at the link layer is evaluated at the transport layer in terms of throughput of the TCP connection.

## II. MODEL OF THE LINK LAYER

The LL receives from the transport layer the data segments to be transmitted, the so-called *Service Data Units* (SDUs). At the LL, a SDU is segmented into  $N$  smaller data units, the *LL data units*, which are stored in a buffer while waiting to be transmitted. Due to error probability of the radio channel, a data unit transmission may fail. In the case of failure, the transmission is repeated until a maximum number,  $M_r$ , of attempts is reached. After  $M_r$  failed transmissions, a LL data unit is discarded and all the other units belonging to the same SDU are removed from the buffer. Thus, at the LL a SDU is lost as soon as one of the LL data units composing the SDU fails  $M_r$  transmissions.

In order to develop a model of the LL, the following aspects have to be considered: the arrival process of SDUs from the upper protocol layers to the link layer, the transmission buffer,

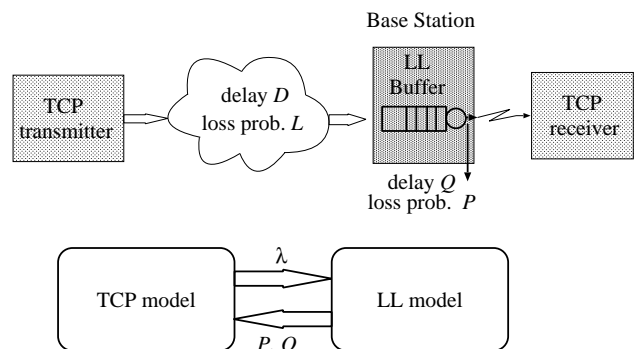


Fig. 1. Sketch of the system under study.

and the radio channel.

The traffic arrival process of the SDUs to the LL is assumed to be Poisson distributed with rate equal to  $\lambda$ . We will discuss later how the arrival process is related to higher layer protocols, and to TCP in particular.

We model the transmission buffer as a M/G/1 queue, in which customers represent SDUs. Assuming that the buffer is properly dimensioned, so that the loss probability due to buffer overflow is negligible, we consider infinite buffer capacity.

In order to accurately describe the SDU service time, taking into account both the SDU segmentation and the channel behavior, we develop a discrete-time Markov chain (DTMC) model of a SDU service. Since a SDU is composed of  $N$  LL data units, the SDU service time consists of  $N$  phases. The duration of a phase depends on the channel status because the transmission of a LL data unit may fail and be repeated. The SDU service time terminates (and the SDU is removed from the buffer) either when all  $N$  data units have been successfully delivered to the receiver (i.e., the  $N$  service phases have been successfully completed) or when one LL data unit is discarded because the maximum number of transmissions has been reached.

The radio channel is modeled as a Gilbert channel [11], with two states, *good* and *bad*, that represent the state of the channel during the transmission time of one LL data unit. The transition probabilities between the two states depend on the steady-state data unit error rate, denoted by  $\epsilon$ , and on the normalized Doppler frequency,  $f_D|\tau|$ . The steady-state data unit error rate can be written as a function of the fading margin  $F$  as follows:  $\epsilon = 1 - e^{-1/F}$ . Clearly, high values of  $F$  represent good channel conditions, while low values of fading margin correspond to a high data unit error rate. The correlation properties of the fading process depend on the normalized Doppler frequency: when  $f_D|\tau|$  is small ( $< 0.1$ ), the process is very correlated (slow fading), while for  $f_D|\tau| > 0.2$  two samples of the process are almost independent (fast fading) [11].

In the DTMC, the time is slotted according to the transmission time of a LL data unit, which is constant and equal to  $\Delta$ . We assume that the sender has knowledge of the transmission outcome of a LL data unit transmission right after its completion. As shown by simulation results (not presented here for the lack of room), this assumption does not have a relevant impact on system performance. We also assume that acknowledgements and negative acknowledgements are always correctly received.

The DTMC state is defined by the vector

$$\bar{s} = (c, r, u)$$

where

- $c$  is the channel state,  $c \in \{good, bad\}$ ;
- $r$  identifies the current transmission attempt for the LL data unit that is under transmission;  $r$  varies between 1 and  $M_r$ ;
- $u$  is the sequence number of the LL data unit that is currently being transmitted;  $u$  can assume all integer values between 1 and  $N$ .

Let  $P(\bar{s}, \bar{d})$  denote the probability that the chain moves in one-step from source state  $\bar{s}$  to destination state  $\bar{d}$ . Probabilities  $P(\bar{s}, \bar{d})$ 's are derived from the behavior of the channel. By employing standard techniques, we compute the steady-state prob-

ability vector  $\bar{\Pi}_L = \{\pi_L(\bar{s})\}$ , where  $\pi_L(\bar{s})$  denotes the steady-state probability of  $\bar{s}$ .

From  $\bar{\Pi}_L$ , we derive the mean and coefficient of variation of the SDU service time,  $E[S]$  and  $C_S$ , respectively. These values are employed in the analysis of the M/G/1 queue. By denoting the traffic intensity by  $\rho = \lambda E[S]$ , the average buffer waiting time  $E[T_w]$  and queueing time  $E[T]$  are given by the Pollaczek-Khintchin formulas,

$$E[T_w] = E[S]\rho \frac{(1 + C_S^2)}{2(1 - \rho)} \quad E[T] = E[T_w] + E[S].$$

An other measure necessary to evaluate the performance of the system is the average service time for the SDUs that are successfully delivered,  $E[S_s]$ . From  $E[S_s]$  we compute the queueing time of a SDU which is not lost as

$$Q = E[T_w] + E[S_s] \quad Q = E[S]\rho \frac{(1 + C_S^2)}{2(1 - \rho)} + E[S_s]. \quad (1)$$

The probability that a SDU is lost at the LL is given by

$$P = \frac{1}{\rho} \sum_{\mathcal{L}} P(\bar{s}, \bar{d}) \pi_L(\bar{s}) \quad (2)$$

where  $\mathcal{L}$  is the set of transitions from  $\bar{s}$  to  $\bar{d}$ , which cause losses.

#### A. Model of the TCP transmitter

In order to evaluate the performance of the TCP connection represented in the upper part of Fig. 1, we employ the Markovian model of TCP NewReno proposed in [12].

The typical bursty nature of many applications which adopt TCP as transport protocol is modeled assuming that the transmitter has an *on/off* kind of behavior. The time spent in each state is a random variable with negative exponential distribution; the mean values are  $T_{off} = 1/\alpha$  and  $T_{on} = 1/\beta$ . When in *off* state, the application is idle; while in *on* state the application has data to transmit and sets up a TCP connection. The behavior of an active TCP connection is described quite in detail in our model introducing states in the CTMC which represent those mechanisms of the protocol having a relevant impact on the performance. In the model development, the TCP window size is measured in segments instead of bytes and segments have all the same size, which is equal to the Maximum Segment Size (MSS). The dynamics of the window size growth are observed with a time granularity equal to the average round-trip time.

Let us focus on the CTMC state space. A part from the *off* state of the application, the state space can be partitioned in three sets. The set  $\mathcal{S}_N$  collects all the states corresponding to the behavior of the TCP transmitter when no loss occurs,

$$\mathcal{S}_N = \{s = (w, W_t)\}$$

where  $w$  denotes the current window size and varies between 1 and the maximum allowed window size  $W_M$ .  $W_t$  represents the window size growth mode (either slow start or congestion avoidance).  $W_t = 1$  stands for congestion avoidance mode; when  $W_t$  is larger than 1 the window grows in slow start mode up to the value  $W_t$ . For simplicity, in slow start mode we only

consider those values of  $W_t$  that the window may take in the exponential growth; i.e., the powers of 2.

The set  $\mathcal{S}_F$  comprises all the states corresponding to the fast recovery and fast retransmit mechanisms which TCP implements in order to recover a loss,

$$\mathcal{S}_F = \{s = (w)\}$$

where  $w$  denotes the window size during fast retransmit and fast recovery procedures.

Finally, states in the set  $\mathcal{S}_T$  describe the protocol waiting for a timeout expiration after a loss occurred,

$$\mathcal{S}_T = \{s = (W_t)\}$$

where  $W_t$  denotes the value assumed by the threshold after the timeout expiration. The state space is then  $\mathcal{S} = \{\text{off}\} \cup \mathcal{S}_N \cup \mathcal{S}_F \cup \mathcal{S}_T$ .

For the sake of brevity we do not provide transition rates between all states, we instead show only an example of transition rate between states in  $\mathcal{S}_N$ . Consider a state  $s = (w, W_t) \in \mathcal{S}_N$ , the window size is equal to  $w$  and the protocol is in slow start mode with threshold equal to  $W_t$ . If none of the  $w$  segments of the window is lost, the window size doubles after roughly a round-trip time. To represent this behavior, the transition rate out of  $s$  has to be equal to the inverse of the average round-trip time and the destination state is  $d = (2w, W_t)$  with probability equal to the probability that no segment is lost. The resulting transition rate from  $s$  to  $d$  is,

$$s = (w, W_t) \in \mathcal{S}_N \rightarrow d = (2w, W_t) \in \mathcal{S}_N : \frac{1}{RTT}(1 - P_t)^w.$$

where  $P_t$  is the total segment loss probability at the transport layer. Similarly, all the other transition rates are computed. For further details see [12].

The steady-state probabilities  $\pi_T(s)$ 's of the CTMC are derived by using standard techniques. From the  $\pi_T(s)$ 's the performance of TCP can be computed. In particular, the average number of segments generated in the time unit, given that the application is *on*, is given by

$$\lambda = \frac{T_{on}}{T_{on} + T_{off}} \sum_{s \in \mathcal{S}_N \cup \mathcal{S}_F} \frac{w}{RTT} \pi_T(s). \quad (3)$$

This value will be used as input in the LL model, as will be explained in the next section.

### B. System performance

The system performance can be derived by combining the LL and TCP models.

The TCP model behavior is determined by the average round-trip time and the segment loss probability; both these values can be derived from the LL model. The TCP model, in its turn, outputs the average generated traffic which can be fed into the LL model. The system behavior is derived by means of a fixed point iterative procedure in which the two models interact by exchanging three values: i) segment loss probability (from LL to TCP), ii) segment delays (from LL to TCP); iii) input traffic at the LL of the wireless link (from TCP to LL). This kind of

iterative approach based on coupling of different models was already used for the study of TCP traffic in a wired environment in [9], [10]. The interaction between the two models is also shown in the lower part of Fig. 1.

Let us now focus on the three exchanged parameters.

The average round-trip time perceived by TCP is given by

$$RTT = 2D + Q + \Delta$$

where  $D$  is the delay due to the wired network,  $Q$  is the queuing delay at the LL as in (1), and  $\Delta$  takes into account the transmission of TCP ACKs over the uplink channel (it is assumed that TCP ACKs do not queue at the LL and are never lost).

Besides losses in the wired portion of the connection, TCP segments are prone to losses in the wireless link; the total segment loss probability  $P_t$  is

$$P_t = 1 - (1 - L)(1 - P)$$

where  $L$  is the probability that the segment is lost in the wired network and  $P$  is given by (2).

We assume that segments arrive at the LL buffer according to a Poisson process, with rate as in (3). Even if the segment generation process at the transmitter is far from being Poisson distributed, when the network is large, the randomness introduced by the network makes the arrival process at the LL buffer closer to being Poisson distributed.

## III. NUMERICAL RESULTS

Results are obtained in the context of 3G wireless systems [13]. According to the 3GPP specifications (Third Generation Partnership Project), while deriving our results we consider that  $\Delta$  is equal to 10 ms and the peak data rate is equal to 384 Kbps. Thus, the LL data unit size results in  $384 \text{ Kbps} \cdot 0.01 \text{ s} = 480$  bytes. We consider a MSS equal to 1,000 bytes, so that each TCP segment is divided into  $N = 3$  LL data units. The TCP maximum window size is equal to 20 segments, and the average *on* and *off* periods of the application are equal to 1 s. We assume that the average delay  $D$  perceived by segments in traversing the network is equal to 100 ms, while we neglect losses due to the wired network.

We first focus on the system performance when the normalized Doppler frequency is constant and equal to 0.1 while the average error probability  $\epsilon$  varies.

Depending on the targeted SDU loss probability which the system intends to guarantee to higher layers, the setting of LL must be adapted to changes in the channel condition. In particular, in order to guarantee a given SDU loss probability, the number of allowed transmission attempts per data unit must increase with the error probability on the channel. This is shown in Fig. 2. Of course, the smaller the loss probability guarantee, the larger the value of  $M_r$ .

The cost of SDU loss probability guarantee is paid in terms of delay perceived by the SDUs. Fig. 3 shows the average SDU queueing delay under the same system conditions as in Fig. 2. For comparison purposes, we also report the throughput that is obtained when the maximum number of transmission attempts at the LL is fixed to 3. In this case, an error probability  $\epsilon = 0.095$  corresponds to a segment loss probability equal to

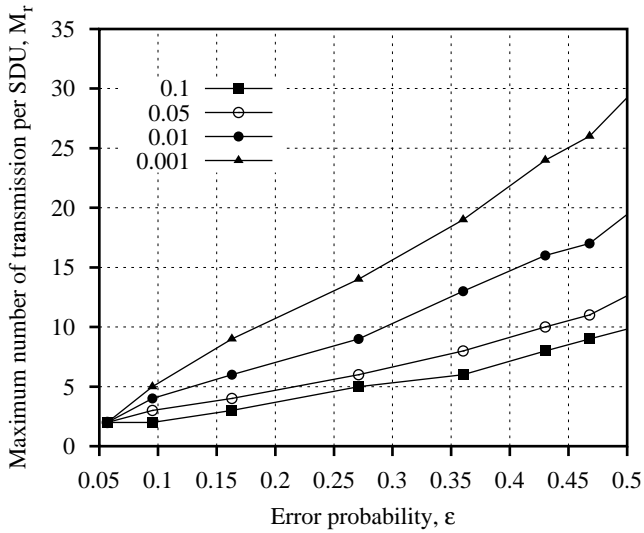


Fig. 2. Maximum number of transmission attempts versus LL data unit error probability for different values of the target SDU loss probability.

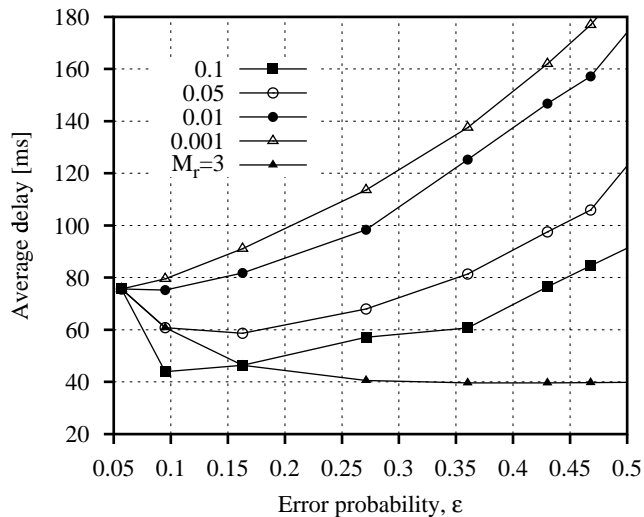


Fig. 3. Mean queueing delay versus LL data unit error probability for different values of the target SDU loss probability and for  $M_r = 3$ .

0.013, while  $\epsilon = 0.45$  corresponds to a loss probability equal to 0.43. Thus, as the channel conditions get worse, the SDU loss probability grows and the average queueing delay decreases.

The effect of LL parameters on higher layer protocols can be observed in Fig. 4, where TCP throughput is shown for different values of error probability  $\epsilon$ .  $M_r$  is again chosen so that the targeted segment loss probabilities can be guaranteed. Parameter settings which correspond to large delay and loss probability make TCP behavior less aggressive and TCP dynamics slower; thus, TCP throughput results smaller. Fig. 4 shows that the TCP throughput that we achieve is much lower than the throughput obtained when the maximum number of transmissions is varied accordingly to the status of the wireless channel.

We now fix the error probability  $\epsilon$  to 0.4 and assess the impact of loss patterns on system performance by letting the normalized Doppler frequency  $f_D|\tau|$  vary. Given  $\epsilon$ , small values of  $f_D|\tau|$  correspond to long bursts of consecutive losses. Fig. 5 shows the value of  $M_r$  needed to guarantee different target SDU loss

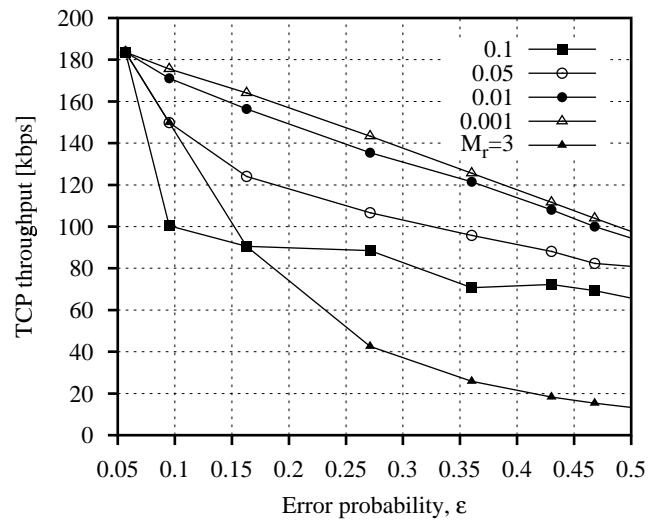


Fig. 4. TCP throughput versus LL data unit error probability for different values of the target SDU loss probability and for  $M_r = 3$ .

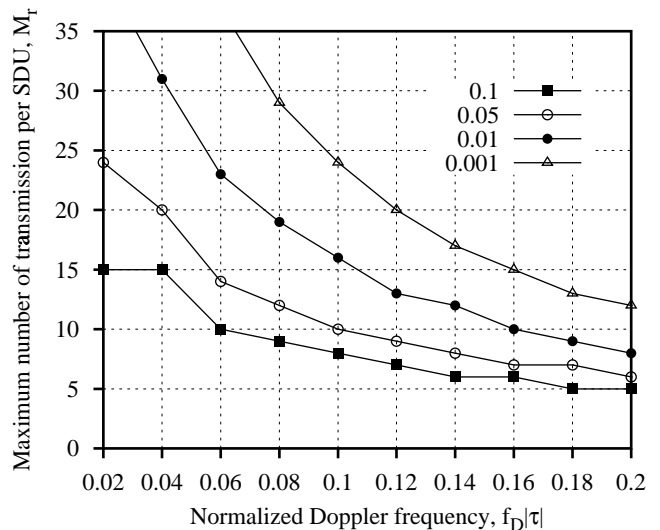


Fig. 5. Maximum number of transmission attempts versus normalized Doppler frequency for different values of the target SDU loss probability.

probability as  $f_D|\tau|$  increases. As expected, loss pattern can have a dramatic impact on system performance; when losses are bursty, large values of  $M_r$  are needed.

Provided that the SDU loss probability is around a target value and given the average error probability  $\epsilon$ , loss pattern does not have a remarkable impact on average SDU queueing delay, which are almost constant as  $f_D|\tau|$  varies (plots are not shown here for the sake of brevity). Loss patterns, instead, influence the coefficient of variation of the service time, as can be seen in Fig. 6.

Finally, Fig. 7 shows the TCP throughput as the normalized Doppler frequency varies, for different targeted segment loss probabilities. Spikes in the two lower curves are due to the fact that different values of  $f_D|\tau|$  are associated the same number of transmission attempts, which correspond to slightly different segment loss probabilities. Observe that TCP throughput does not vary significantly as  $f_D|\tau|$  changes. In fact, TCP throughput is mainly determined by average losses and delays. Choosing  $M_r$  so that SDU loss probability is guaranteed below a given

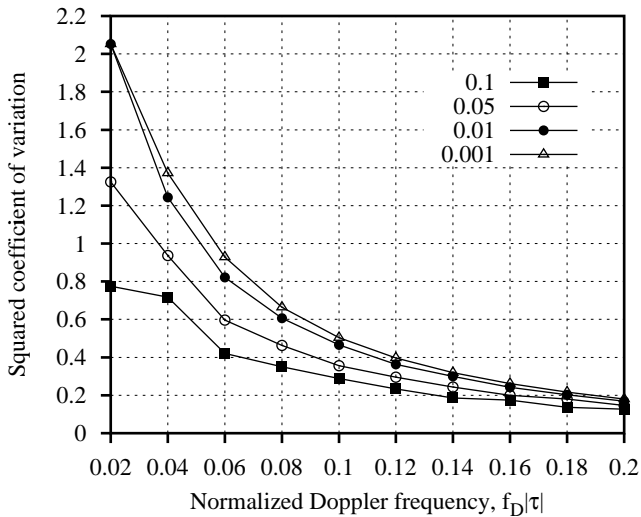


Fig. 6. Squared coefficient of variation of queuing delay versus normalized Doppler frequency for different values of the target SDU loss probability.

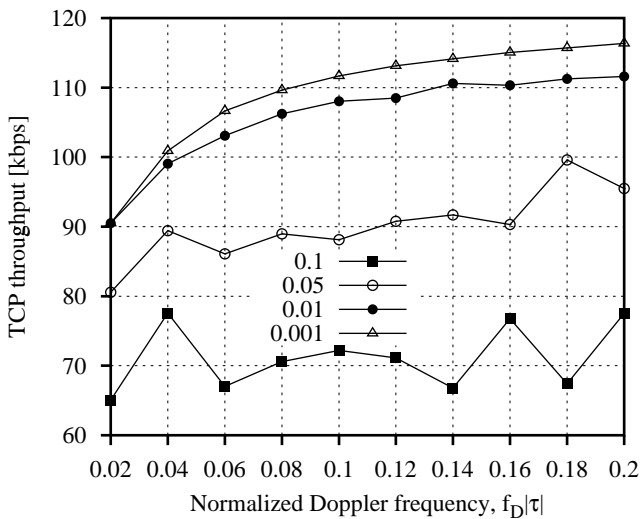


Fig. 7. TCP throughput versus normalized Doppler frequency for different values of the target SDU loss probability.

target value, both losses and average delay result to be almost constant, and TCP throughput does not vary.

#### IV. CONCLUSIONS AND FUTURE WORK

We studied the performance of TCP when the last hop of the end-to-end connection is wireless and link-layer retransmissions are used to shield the TCP sender from losses over the wireless channel. By using Markovian models of the link layer error-recovery scheme and of TCP, we determined the link-layer parameter setting which provides the desired QoS at the transport layer. The analysis was carried out for different values of the error probability and of the correlation of the fading process over the wireless channel. Results showed that by properly selecting the maximum number of transmission attempts at the link layer as the fading error probability varies, the TCP throughput greatly increases with respect to the case where a fixed number of transmissions attempts is performed. Moreover, fixed the fading error probability, constant TCP throughput can be

achieved as the correlation of the fading process varies, by setting the maximum number of link-layer transmissions at a suitable value.

Further research will focus on the study of the TCP performance when service disruptions due to handoff procedures are taken into account and services with different values of bit-rate are considered. Also, the possibility to exploit link layer error-recovery functions to satisfy QoS constraints for other kind of applications will be explored.

#### REFERENCES

- [1] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, and R.H. Katz, "A comparison of mechanisms for improving TCP performance," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 756–769, December 1997.
- [2] A. Chockalingam, M. Zorzi, and V. Tralli, "Wireless TCP performance with link layer FEC/ARQ," in *Proc. of ICC*, 1999, pp. 1212–1216.
- [3] J.W.K. Wong and V.C.M. Leung, "Improving end-to-end performance of TCP using link-layer retransmissions over mobile internetworks," in *Proc. of ICC*, 1999, pp. 324–328.
- [4] F. Anjum and R. Jain, "Performance of TCP over lossy upstream and downstream links with link-level retransmissions," in *Proc. of ICON*, 2000, pp. 3–7.
- [5] Y. Bai, A.T. Ogielski, and G. Wu, "Interactions of TCP and radio link ARQ protocol," in *Proc. of IEEE VTC*, 1999, pp. 1710–1714.
- [6] E. Ayanoglu, S. Paul, and *et al.*, "A link-layer protocol for wireless networks," *ACM/Baltzer Wireless Networks Journal*, vol. 1, pp. 47–60, February 1995.
- [7] Chaskar N.M., T.V. Lakshman, and U. Madhow, "TCP over wireless with link level error control: analysis and design methodology," *IEEE/ACM Transactions on Networking*, vol. 7, no. 5, pp. 605–615, October 1999.
- [8] J.H. Hu and K.L. Leung, "Hierarchical cache design for enhancing TCP over heterogeneous networks with wired and wireless links," in *Proc. Globecom 2000*, November 2000.
- [9] C. Casetti and M. Meo, "A new approach to model the stationary behavior of TCP connections," in *Proc. of INFOCOM 2000*, Tel Aviv, Israel, March 2000.
- [10] T. Bu and D. Towsley, "Fixed point approximation for TCP behavior in an AQM network," to appear in *Proc. of ACM SIGMETRICS 2001*.
- [11] M. Zorzi, R.R. Rao, and L.B. Milstein, "ARQ error control for fading mobile radio channels," *IEEE Transactions on Vehicular Technology*, vol. 46, no. 2, pp. 445–55, May 1997.
- [12] C. Casetti and M. Meo, "Modeling the stationary behavior of TCP Reno connections," in *Proc. of International Workshop on QoS in Multiservice IP Networks*, Rome, Italy, January 2001.
- [13] Third Generation Partnership Project (3GPP), "Technical specifications," March 2000, <http://www.3gpp.org>.