On the Effects of ARQ Mechanisms on TCP Performance in Wireless Environments

Francesco Vacirca, Andrea De Vendictis, Alfredo Todini, Andrea Baiocchi
INFOCOM Dept. - University of Roma *"La Sapienza"*
Via Eudossiana 18, 00184, Roma (Italy)
{vacirca,devendictis,todini,baiocchi}@infocom.uniroma1.it

***Abstract*— In this paper we investigate the interaction between TCP and wireless ARQ mechanisms. The aim is to understand what is the best reliability degree of the wireless link in order to guarantee TCP performance. For this purpose, we first develop a Markov model for a selective repeat ARQ protocol, widely used in the current wireless environments. Secondly, we design a cross-layer algorithm that, by exploiting the proposed model, can adapt the number of link layer transmission attempts to the end-to-end packet loss rate perceived by TCP. The interaction between TCP and link layer is evaluated in a specific case study (TCP over 3G radio access) by means of simulations carried out by using a very detailed UMTS-TDD simulator based on *ns*. The deployment of the link layer Markov model and of the proposed algorithm allows us to derive some interesting conclusions about the design of retransmission protocols in TCP/IP network environments.**

I. Introduction

In the last years, the issues regarding the behavior of TCP over wireless links have been extensively addressed in the networking research community, both because TCP represents the most widespread transport protocol in the current Internet (about 90 percent of the overall IP traffic is carried by TCP according to recent experimental measurements [1]) and because Internet access through wireless technologies is rapidly taking up. Recent and thorough reviews of general problems and possible solutions regarding this specific topic can be found for example in [2], [3] and [4].

In this context, the specific aim of the present work is to investigate the impact of wireless link layer ARQ mechanisms on TCP performance through both analytic methods and simulation environments. In particular we are interested in studying how the degree of persistence of the radio link layer affects end-to-end TCP performance and wireless link utilization.

In the current literature this aspect of the link layer design appears to be quite a controversial issue. In fact, with respect to the three commonly considered solutions - persistent and fully reliable ARQ (i.e., unlimited number of retransmission attempts), semi-reliable ARQ (i.e., with a fixed limit on the number of retransmissions), and ARQ with a dynamically adaptive limit - there is no agreement on which of them is the best in order to maximize TCP performance.

Most of the works on TCP over wireless (e.g., [5], [6]) implicitly assume that wireless links are not fully shielded from "random" errors caused by the non-ideality of the radio channel (i.e., they assume an unreliable or semi-reliable link layer); therefore they propose a number of mechanisms in order to avoid undesired reductions of TCP throughput in presence of "non-congestion" wireless packet losses. A non persistent ARQ approach would have the advantage of reducing spurious TCP timeouts and packet reordering with a benefit for TCP.

Conversely, another "school of thought" (e.g., [7], [8]) asserts that the wireless link layer should be fully reliable to preserve TCP performance. This choice guarantees that any loss detected by TCP is due to congestion.

In [9] the authors propose a link layer algorithm that adapts the number of retransmission attempts according to a target loss rate, used as parameter to describe a desired QoS for a TCP connection. They show that TCP performance can be improved by adapting the maximum number of retransmissions to the desired QoS rather than leaving it constant.

In a previous work [10] it has been verified by means of simulations that, when the packet loss probability experienced by the TCP connection in the wireless link is of the same order of magnitude as the one experienced in the wired network portion, TCP performance is essentially not affected by the wireless impairment. Hence, we can argue that TCP performance is not substantially improved by implementing a fully reliable wireless link layer, when TCP experiences significant packet loss events in the wired part of the network. Moreover, a number of earlier experimental works (see for example [11] and [12]) pointed out that the wired Internet is far from being ideal, given that congestion packet loss events are rather frequent (even 1 to 10 percent). To force a large number of retransmissions in the wireless link could have a negative impact in terms of utilization efficiency, without improving TCP performance.

In this work, we develop a Markov model able to reproduce a selective repeat ARQ link layer protocol, widely used in the wireless link environments (e.g., 3G systems). By means of this model, by adopting an approach similar to the one developed in [9], we design an adaptive cross-layer algorithm that dynamically varies the number of maximum retransmissions in the ARQ protocol according to the end-to-end loss rate experienced by TCP. The novelty with respect to [9] is that the target loss rate of the link layer is actually derived from TCP running estimates of the end-to-end packet loss. We evaluate the analytic model and the proposed algorithm in a specific and realistic case study (TCP over UMTS radio access) by means of a very detailed *ns* based UMTS-TDD simulator.

The analytic link layer model and the evaluation process of the proposed adaptive algorithm allow us to derive some interesting conclusions about the impact of link layer retransmission mechanisms on TCP performance.

The rest of the paper is structured as follows. In Section II we depict the network reference scenario. In Section III we describe the adaptive retransmission algorithm and the link layer Markov model. In Section IV we evaluate the proposed algorithm through a number of simulations and discuss the results. Finally we give the conclusions and some hints to further work.

 0-7803-7974-8/03/$17.00 © 2003 IEEE

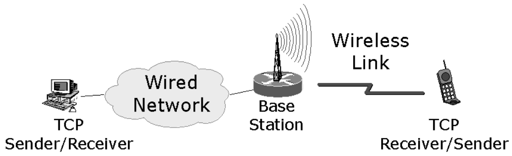

Fig. 1. Network scenario.

II. Reference Network Scenario

The reference network scenario considered in this work is depicted in Figure 1. A TCP sender (receiver) placed on a mobile node establishes a connection with a TCP receiver (sender) placed within the wired section of the Internet. We use the TCP Reno version [14], today the most popular TCP implementation. The TCP source is a "greedy source", always ready to send new data. The rapid development of "peer-to-peer" applications and high-speed access links, that allow the exchange of large amount of data among users, makes this hypothesis rather realistic. Besides, already now, according to [1], streams having lifetimes of hours to days carry a high proportion (50 or 60 percent) of the total amount of traffic on the wired Internet.

In order to reproduce the non-ideality of the network, TCP packets are dropped in the wired portion of the Internet according to a random process assumed independent of the TCP connection status[1]. The wireless access network of the mobile TCP terminal is assumed to be a UMTS-TDD radio interface, although for the purpose of this work it could be a generic wireless link implementing a selective repeat ARQ mechanism.

Here we only describe the acknowledged mode (AM) of the UMTS RLC layer ([13]) with the options used in our study, given that a detailed presentation of the UMTS-TDD system is beyond the purpose of this paper.

The AM RLC uses a selective repeat ARQ mechanism for error recovery. Data of arbitrary size is accepted from the upper layers and, once segmented into fixed size PDUs, placed in the RLC transmission buffer; PDUs are transmitted when the MAC indicates that there are available transmission resources. A sliding window mechanism regulates the transmission of PDUs. On reception of a PDU, after decoding, the physical layer performs error detection, and passes the result of the check to the RLC layer, together with data. When the last PDU of a SDU is received, the receiving RLC sends a Status PDU to the transmitter to indicate which PDUs were received correctly and which PDUs require retransmission. To avoid deadlock states a number of timers are implemented in the protocol.

Many parameters can be configured; among them the size of the transmission window, the size of Data PDU and Status PDU, the size of the RLC transmission buffer and the duration of the timers. Of particular relevance to us is the possibility to set the maximum number of retransmissions per each PDU: when this number is reached, the RLC determines that it is impossible to deliver the data correctly, therefore it discards the whole SDU and advertises the peer RLC entity. The RLC has been configured for in-sequence delivery.

[1]This hypothesis appears to be valid when the traffic generated by the target TCP connection is a negligible fraction of the overall traffic crossing the TCP connection path in the considered wired network.

III. Adaptive Link Layer Algorithm

By following a cross-layer approach, the proposed adaptive algorithm is based on the exchange of information between the wireless link layer and the TCP entity (either sender or receiver) placed on the mobile node.

Let P_{TCP} be the TCP end-to-end packet loss probability, P_W be the probability of packet losses due to congestion and P_{LL} be the TCP packet loss probability on the wireless link[2].

If we assume that P_W and P_{LL} are independent, the following expression holds:

$$1 - P_{TCP} = (1 - P_W)(1 - P_{LL}) \tag{1}$$

Given P_{LL} and P_{TCP}, we can derive P_W from (1). Hence, we compute the *target* link layer packet loss probability $\hat{P}_{LL}$ as $\hat{P}_{LL} = \alpha P_W$, with $0 \leq \alpha \leq 1$ a system parameter representing the percentage amount of losses we accept in the wireless link with respect to the losses due to congestion. The key idea is to adapt the link layer to balance the "random" packet losses experienced in the wireless link to the losses due to congestion.

Given the *target* link layer packet loss probability $\hat{P}_{LL}$, we are able to compute the maximum number of retransmission attempts by using the expression derived from the analytic model developed in the following Section III-A.

The packet loss probability P_{TCP} experienced on the end-to-end path and the packet loss probability P_{LL} experienced on the wireless section are estimated by means of the moving window algorithms described in Section III-B.

A. Computing the maximum ARQ retransmission attempts

In this section we describe the discrete Markov model developed to reproduce the reference wireless link environment depicted in Section II and to derive the maximum retransmission attempts of the ARQ protocol, given a *target* link layer SDU loss probability[3].

The radio channel is modeled as a Gilbert-Elliot channel with two states (GOOD and BAD) [15], widely used in related literature to characterize wireless transmission medium (see for example [16]). The GOOD state is the one with very low packet error probability p_G, whereas the BAD state is the one with high packet error probability p_B.

The discrete Markov chain, with unit time τ, is characterized by the following transition probabilities: g and b represent the probability that the channel remains respectively in the state GOOD and BAD during the sampling time τ and $1-g$ and $1-b$ the transition probabilities from the GOOD state to the BAD state and vice-versa. The b and g parameters can be estimated by direct field measurements on the basis of the mean time in the BAD state $\tau/(1-b)$ and of the mean time in the GOOD state $\tau/(1-g)$.

The selective repeat ARQ protocol is modeled with a discrete Markov chain which reproduces the transmission of a single and isolated SDU, under the following assumptions: i) the SDU is segmented at the link layer in n_{pdu} PDUs of equal size; ii) all

[2]By TCP packet loss on the wireless link we mean the residual packet error after the deployment of link layer ARQ mechanisms. P_W includes both the losses due to RLC buffer overflows and the losses experienced in the wired network.

[3]A link layer SDU corresponds to an IP packet carrying a TCP segment.

 0-7803-7974-8/03/$17.00 © 2003 IEEE

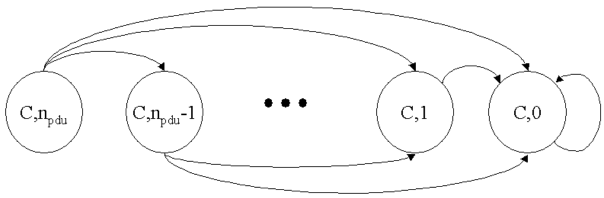

Fig. 2. Markov chain.

the PDUs belonging to the SDU are transmitted in a single time interval of duration τ; iii) the ack is delivered without errors at the end of the time interval; iv) all the PDUs which require retransmission are sent again in the next time interval, and so on; v) the error events of different PDUs conditional on the state of the channel are independent of one another.

A chain state is defined by the vector $\overline{s} = (C, X)$, where:

- $C \in \{\text{GOOD}, \text{BAD}\}$ is the Gilbert-Elliot channel state;
- $X \in [0, n_{pdu}]$ is the number of PDUs, belonging to the considered SDU, still to be transmitted.

In Figure 2 the Markov chain for a single state channel C is shown. A transition from the state (C, k) to the state (C, k') represents the event that $k - k'$ PDUs are transmitted successfully and k' PDUs are transmitted with error; only transitions from k to k' with $k' \leq k$ are allowed. The transition probability $b_C(k, k')$ is the probability to fail k' transmissions out of k:

$$b_C(k, k') = \binom{k}{k'} p_C^{k'} \cdot (1 - p_C)^{k-k'}$$

where p_C is the error probability in the state C. The transition matrix $\mathbf{T}_C$ is:

$$\mathbf{T}_C = \begin{bmatrix} 1 & 0 & \dots & 0 \\ b_C(1,0) & b_C(1,1) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_C(n_{pdu},0) & b_C(n_{pdu},1) & \dots & b_C(n_{pdu},n_{pdu}) \end{bmatrix}$$

Extending the chain for the two-state Gilbert-Elliot channel, the transition probability matrix of the whole process becomes:

$$\mathbf{P} = \begin{bmatrix} g \cdot \mathbf{T}_G & (1-g) \cdot \mathbf{T}_G \\ (1-b) \cdot \mathbf{T}_B & b \cdot \mathbf{T}_B \end{bmatrix}$$

Let $\mathbf{T}_C^*$ be a matrix obtained from $\mathbf{T}_C$ by removing the first row and the first column, the matrix $\mathbf{Q}$ is defined as:

$$\mathbf{Q} = \begin{bmatrix} g \cdot \mathbf{T}_G^* & (1-g) \cdot \mathbf{T}_G^* \\ (1-b) \cdot \mathbf{T}_B^* & b \cdot \mathbf{T}_B^* \end{bmatrix}$$

$\mathbf{Q}$ is the one step transition probability matrix of a transient Markov chain, whose absorption time η gives the number of frames needed to complete the delivery of the PDUs making up an SDU. The residual SDU loss probability, provided the number of transmission attempts per PDU is n, is defined as follows: $F(n) = Pr\{\eta > n\} = \boldsymbol{\pi}\mathbf{Q}^n\mathbf{e}$ for $n \geq 0$ where $\mathbf{e} = [1 \dots 1]^T$ and $\boldsymbol{\pi}$ is the $2n_{pdu}$ components row vector of the initial probabilities; $\boldsymbol{\pi}$ is equal to $[0 \dots \pi_G, 0 \dots \pi_B]$, where π_C is the initial probability to stay in the channel state C.

The expected value of η is the mean number of transmissions per SDU and can be evaluated as follows:

$$E[\eta] = \sum_{n=0}^{\infty} n Pr\{\eta = n\} = \sum_{n=0}^{\infty} F(n) = \boldsymbol{\pi}(\mathbf{I} - \mathbf{Q})^{-1}\mathbf{e} \quad (2)$$

The mean number of transmissions per SDU, when the number of transmissions per PDU is limited to M_t, is:

$$E[\eta | n \leq M_t] = \sum_{n=0}^{M_t - 1} F(n) = \boldsymbol{\pi}(\mathbf{I} - \mathbf{Q}^{M_t})(\mathbf{I} - \mathbf{Q})^{-1}\mathbf{e} \quad (3)$$

When n_{pdu} is equal to 1, equations (2) and (3) represent the mean value of the number of transmissions per PDU.

In Section IV we use the ratio $\mathcal{R} = E[\eta | n \leq M_t]/E[\eta]$ computed for $n_{pdu} = 1$ to explain the impact of retransmission attempts on the wireless link efficiency.

Now we derive a useful approximate expression for the residual SDU loss probability $F(n)$. $\mathbf{Q}$ is similar by row and column permutations to a triangular block matrix with blocks of 2×2 elements; the eigenvalues can be easily computed from the diagonal blocks. It is possible to show that the maximum eigenvalue of $\mathbf{Q}$ is λ_1 with $0 < \lambda_1 < 1$:

$$\lambda_1 = \frac{gp_G + bp_B + \sqrt{(gp_G + bp_B)^2 - 4p_B p_G(g + b - 1)}}{2}$$

Furthermore, from the spectral decomposition of $\mathbf{Q}$, the following limit holds:

$$\lim_{n \to \infty} \frac{\mathbf{Q}^n}{\lambda_1^n} = \mathbf{u}_1\mathbf{v}_1 \quad (4)$$

where $\mathbf{u}_1$ and $\mathbf{v}_1$ are the right and the left eigenvectors of $\mathbf{Q}$ relevant to λ_1, normalized so that $\mathbf{v}_1\mathbf{u}_1 = 1$.

Hence, we can approximate $F(n)$ by using the asymptotic value from the equation (4) and write:

$$F(n) \simeq \boldsymbol{\pi}\mathbf{u}_1\mathbf{v}_1\mathbf{e}\lambda_1^n = c\lambda_1^n$$

To find the constant c we must calculate the eigenvectors of the matrix $\mathbf{Q}$ belonging to λ_1. The eigenvectors computation complexity can be too high for a run-time algorithm. Therefore in our link layer mechanism we adopt a further approximation of $F(n)$, named $\hat{F}(n)$. $\hat{F}(n)$ is obtained by imposing that $\hat{F}(1)$ is the error probability without retransmissions $F(1) = c \cdot \lambda_1 = 1 - [\pi_G(1 - p_G)^{n_{pdu}} + \pi_B(1 - p_B)^{n_{pdu}}]$ and assuming $\hat{F}(n)$ has the same asymptotic slope of $F(n)$. Therefore, for $n \geq 1$:

$$\hat{F}(n) = \left[1 - [\pi_G(1 - p_G)^{n_{pdu}} + \pi_B(1 - p_B)^{n_{pdu}}]\right]\lambda_1^{n-1} \quad (5)$$

The maximum number of transmissions per PDU M_t is set by inverting (5), given a requirement on $F(M_t)$.

The Markov model has been validated by means of simulations carried out with a UMTS-TDD module for *ns* that simulates the TD-CDMA radio interface shown in Section II[4]. In all the simulations shown in this section and in Section IV, the following parameters are left unchanged: TCP packet size is

[4]The UMTS module, available at [19], is able to accurately reproduce physical, MAC and RLC layers of UMTS-TDD.

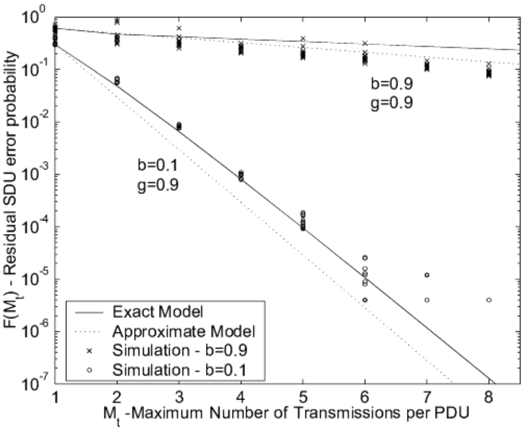

Fig. 3. Selective repeat ARQ Markov model validation.

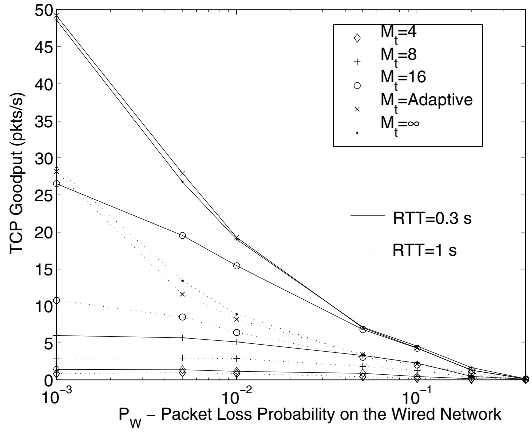

Fig. 4. TCP Goodput varying the packet loss probability on the wired network.

1000 bytes, the RLC payload size is 40 bytes (so that the number n_{pdu} of PDUs per SDU is 25), the error probabilities in the GOOD and BAD states are respectively $p_G = 0.01$ and $p_B = 0.9$, the capacity of the links in the wired Internet is large enough to avoid buffer backlogs.

In Figure 3 the loss probability values computed by the model (solid line), by the approximate expression (5) (dotted line) and as derived from the simulations (circles and crosses) are shown for $g = 0.9$ and $b = 0.1$ and 0.9, corresponding to a low error burstiness and high error burstiness respectively.

Given the accuracy of the simulator in reproducing UMTS stack, several hypotheses necessary to define the model are not verified in the simulations (e.g., PDUs are no longer independent, the reception of the STATUS PDU is not instantaneous, timers and the sliding window are implemented in the simulator). In spite of these differences and of the fact that we use TCP as transport protocol, both exact and approximate models appear to be quite accurate against the simulation results.

B. Estimating packet loss probabilities

In order to compute the end-to-end TCP packet loss probability P_{TCP} we first have to detect TCP packet loss events. We consider two cases: i) as for the TCP receiver we use the algorithm described in [17]. The algorithm detects a lost packet by the arrival of at least three packets with a higher sequence number than the lost packet; that to be coherent with the TCP fast retransmit algorithm; ii) as for the TCP sender, the packet loss detection is driven by fast retransmit and timeout events.

To compute the TCP loss event probability P_{TCP}, we calculate the average number of packets I_{mean} successfully transmitted between two consecutive loss events (called *loss event intervals*). This is done using a filter that weights the eight most recent loss event intervals in such a way that the measured loss event probability changes smoothly. The filter developed to derive I_{mean} is throughly described in [17]. Thus, the TCP loss event probability is $P_{TCP} = 1/I_{mean}$.

TCP passes the value of P_{TCP} to the link layer entity after every update. If the TCP sender is placed on the mobile node, P_{TCP} is used directly by the link layer entity below; otherwise, if the TCP receiver is placed on the mobile node, P_{TCP} is used by the link layer entity placed on the base station.

As for the wireless packet loss probability P_{LL} estimation, the RLC entities have to recognize two events: i) the correct reception of a SDU; ii) the loss of a SDU. The first event is detected when the RLC transmitting entity receives a Status PDU acknowledging (or completing the acknowledgment of) all the PDUs which make up an SDU. The second event is detected when, after reaching the maximum number of retransmissions, the RLC layer drops an SDU.

To estimate the error probability, we compute a moving average of the loss frequency over a suitable number of transmitted SDUs (in the simulations we use a window of 100 SDUs).

IV. RESULTS

In this section we discuss the results of the simulations carried out to investigate the interaction between different persistent degree RLC protocols and TCP. The simulation scenario is the one described in Section II.

First, to point out the effect of limiting the maximum number of retransmission attempts, the TCP goodput as function of the wired packet loss probability is shown in Figure 4 for different values of M_t, for M_t unlimited and for M_t adaptive according to the proposed algorithm. In these simulations the round trip delay on the wired section of the network is 0.3 s (solid line) and 1 s (dotted line), $\alpha = 0.1$, $b = 0.9$, $g = 0.9$ and the RLC buffer is assumed infinite so that the only losses on the radio interface are due to the channel impairment and P_W coincides with the wired loss probability.

In Figure 5 the TCP goodput is shown as function of the RLC buffer size when M_t is unlimited and when the adaptive algorithm is implemented (for the values of α 0.1 and 1). In this case we assume no losses in the wired network.

As shown in Figure 4, the maximum TCP goodput is always obtained either with unlimited M_t or by adapting M_t according to the proposed algorithm. We observe a degradation in TCP performance when M_t is limited independently of the loss experienced by TCP in the wired network. By using the adaptive algorithm we introduce residual loss probability on the wireless link ($\alpha = 0.1$ corresponds to a loss probability 10 percent of the wired loss probability), nevertheless TCP performance is

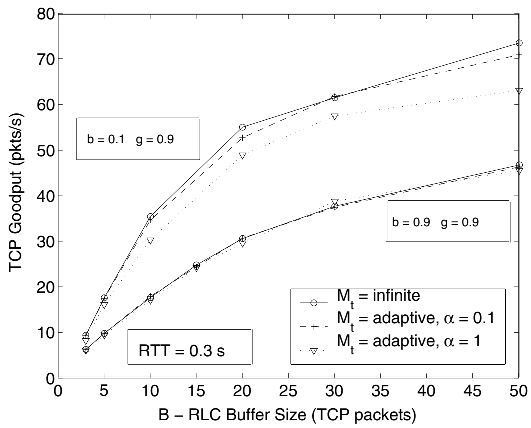

Fig. 5. TCP Goodput varying the RLC buffer size B.

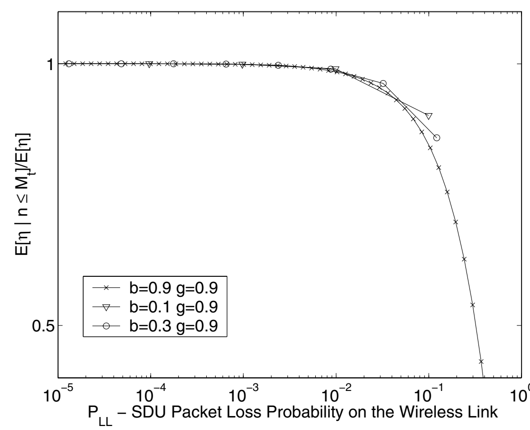

Fig. 6. $E[\eta|n \leq M_t]/E[\eta]$ varying the wireless loss probability P_{LL}.

not affected. This is also confirmed by the results shown in Figure 5, where it is possible to note that TCP performance is not affected by the residual error rate introduced by the algorithm, in particular for high error burstiness ($b = 0.9$) and $\alpha = 0.1$.

The aim of the proposed adaptive algorithm would be to improve wireless link utilization without reducing TCP performance. As shown in the previous figures our algorithm satisfies the condition upon TCP performance; however, it gives no improvement in terms of wireless link efficiency as shown in Figure 6, which graphs the ratio $\mathcal{R}$ derived in Section III-A. The value of $\mathcal{R}$ represents the gain margin in terms of efficiency that can be obtained by setting the maximum number of transmissions (M_t) with respect to the case of unlimited M_t. To obtain an improvement in wireless efficiency, $\mathcal{R}$ should be substantially less than 1. In the figure it appears that $\mathcal{R}$ begins to decrease only for very high loss probabilities (larger than 0.01) on the wireless link. This behavior is confirmed by computing the energy efficiency (i.e., the ratio between the number of bits correctly received and the bits transmitted) in the simulations with fixed M_t: for instance, when $RTT = 0.3\ s$ and $b = 0.9$, the energy efficiency decreases from 0.65 to 0.48 as M_t increases from 4 to 16, whereas it does not change as M_t increases from 16 to 32, while TCP goodput continues to increase.

So the ranges of values for which wireless link efficiency could be improved by limiting the maximum number of retransmission attempts is not feasible from the point of view of TCP performance. This holds for our algorithm as well as for any algorithm that confines itself to (either statically or adaptively) limiting the number of retransmission attempts.

Summing up, on the one hand, for TCP performance it is not strictly necessary to provide unlimited retransmissions in the wireless link (in fact, our adaptive algorithm achieves equal performance by limiting the retransmissions). On the other hand, there is no improvement in wireless efficiency if we limit the number of retransmissions. Hence, the best and simplest choice for TCP is to adopt a fully reliable link layer protocol. In the light of these facts, designing ad hoc TCP protocols that try to distinguish "congestion" losses from "wireless" losses seems to be pointless.

V. Conclusions

In this paper we investigated the interactions between ARQ mechanisms and TCP behavior. The major contribute of this work is to give some precise indications on the design of the link layer protocol for mobile computing applications.

As further work we want to verify our conclusions in network scenarios including multiple and non-persistent TCP flows.

References

[1] N. Brownlee, K.C. Claffy, *Understanding Internet Traffic Streams: Dragonflies and Tortoises*, IEEE Communications Magazine, October 2002.

[2] G. Xylomenous et al., *TCP Performance Issues over Wireless Links*, IEEE Communications Magazine, April 2001.

[3] M. Allman, Editor, *Ongoing TCP Research Related to Satellites*, RFC2760, IETF, February 2000.

[4] C. Barakat et al., *On TCP Performance in a Heterogeneous Network: A Survey*, IEEE Communications Magazine, January 2000.

[5] C. Casetti et al., *TCP with Faster Recovery*, Proc. of IEEE MILCOM'2000, Los Angeles,CA, USA, September 2000.

[6] C. P. Fu, S. C. Liew, *TCP Veno: TCP Enhancement for Transmission over Wireless Access Networks*, IEEE Journal on Selected Areas in Communications, Vol. 21, No. 2, February 2003.

[7] H. M. Chaskar et al., *TCP Over Wireless with Link Level Error Control: Analysis and Design Methodology*, IEEE/ACM Trans. on Networking, Vol.7, No.5, October 1999.

[8] R. Ludwig, *A Case for Flow-Adaptive Wireless Links*, University of California at Berkeley, Technical Report UCB/CSD-99-1053, May 1999.

[9] C.F. Chiasserini, M. Meo, *A Reconfigurable Protocol Setting to Improve TCP over Wireless*, IEEE Transactions on Vehicular Technology, 2002.

[10] F. Vacirca, A. Baiocchi, *End-to-End Evaluation of WWW and File Transfer Performance for UMTS-TDD*, Proc. of IEEE Globecom'02, Taipei (Taiwan), November 2002.

[11] Vern Paxson, *End-to-End Internet Packet Dynamics*, IEEE/ACM Transactions on Networking, Vol. 7, no. 3, June 1999.

[12] A. De Vendictis, A. Baiocchi, *Wavelet Based Synthetic Generation of Internet Packet Delays*, Proc. of ITC'17, Salvador (Brazil), December 2001.

[13] TS25.322 *Radio Link Control (RLC) Protocol Specification*, Release 5 v5.1.0, June 2002.

[14] R. W. Stevens, *TCP/IP Illustrated,Vol I The protocols*, Addison-Wesley, U.S.A., 1994.

[15] E.N. Gilbert, *Capacity of Burst Noise Channels*, The Bell System Technical Journal, Vol. 39, pp. 1253-1256, 1960.

[16] M.Zorzi et al.,*On the Accuracy of a First-order Markov Model for Data Transmission on Fading Channels*, Universal Personal Communications. 1995. Record., IEEE International Conference on , November 1995.

[17] Mark Handley et al., *TCP Friendly Rate Control (TFRC): Protocol Specification*, IETF draft, October 2002.

[18] ns-LBL network simulator ns-2.1b9a, documentation and software available via http://www.isi.edu/nsnam/ns/index.html.

[19] A. Todini, F. Vacirca, *UMTS module for ns*, available via http://net.infocom.uniroma1.it.