Improving TCP over Wireless by Selectively Protecting Packet Transmissions

Carla F. Chiasserini, Michele Garetto, Michela Meo
Dipartimento di Elettronica, Politecnico di Torino
Corso Duca degli Abruzzi 24, 10129 Torino, Italy
E-mail:{chiasserini,garetto,michela}@polito.it

***Abstract*—One of the major challenges in modern communication systems is to provide wireless access to Internet. Since TCP is expected to be used for Internet connectivity at the transport layer, we need to improve TCP performance over wireless links. Our key idea is to identify the TCP data segments which are of 'critical' importance to the protocol evolution, and, by using FEC coding, increase the probability of successfully transmitting them over the radio channel. The performance of the proposed approach is evaluated in terms of file transfer completion time.**

***Index Terms*— TCP over wireless, Wireless networks, Error-recovery, FEC/ARQ, Performance evaluation, QoS**

I. Introduction

Enabling mobile users to access the vast range of Internet-based applications is the next step in wireless communications evolution. The *de facto* standard for Internet connectivity is represented by the TCP/IP protocol suite. TCP performance, however, may significantly degrade when end-to-end connections include wireless links because TCP considers losses due to the radio channel as signs of network congestion. In this case, TCP invokes congestion control mechanisms which are intended to reduce the connection throughput and increase the end-to-end data transfer delay [1], [2], [3].

Several proposals to provide packet traffic with the desired quality of service (QoS), in spite of the disadvantageous radio propagation conditions have appeared in the literature [4], [5], [6], [7], [8], [9]. In particular, in [8], [9], ARQ (Automatic Repeat reQuest) and FEC (Forward Error Correction) schemes have been used to obtain local data reliability and make the wireless link appear to TCP as a reliable one, although with a longer delay.

In this paper, we devise a novel technique that, besides employing ARQ, makes use of a FEC scheme to *selectively* protect those TCP packets that are of 'critical' importance to the protocol performance.

As sketched in Fig. 1, the critical TCP segments are identified at the TCP transmitter by properly setting the TOS field of the IP header. Segments are then carried in the wired network, where intermediate routers will typically ignore the TOS field. Once the segments arrive at the Base Station in charge of the last wireless hop of the connection, the TOS field is interpreted at the link layer of the protocol stack. If the segment is identified as a critical one, a FEC scheme is applied in order to make the segment transmission over the radio channel more reliable.

We highlight that while applying FEC coding to all the packets allows for a higher data transfer reliability, it causes a significant waste of bandwidth and energy resources when channel conditions are fairly good. On the contrary, a selective scheme can achieve a convenient trade-off between energy saving and performance.

Different schemes can be proposed based on the choice of the critical packets to be protected. In what follows, the proposed scheme is tailored to improve the performance of short lived flows.

The presented approach preserves the end-to-end TCP semantics and involves only a marginal modification to the TCP implementation at the transmitter side.

II. Selective Protection of TCP Packets

We focus on TCP NewReno, although other versions of the protocol can be considered as well.

Since completion time is a key metric of the QoS perceived by end users, we focus on file transfer completion time as objective of our TCP performance improvement. In terms of completion time, the impact of a segment loss is not always the same. In fact, the time necessary to recover a lost segment depends on the position of the segment within the current transmission window, on the state of the protocol, and on previous losses. Our idea is to use FEC coding to protect those segments whose loss would significantly degrade the QoS, i.e., the completion time.

TCP infers that a segment is lost whenever one of the following two events occurs: retransmission timeout (RTO) expires, or three duplicate ACKs trigger the Fast Retransmit mechanism. Of these two events, RTO is the most undesirable one as the RTO period is usually much larger than the time necessary to recover a loss by Fast Retransmit, which is about equal to a round trip time. Therefore, the proposed selective protection scheme aims at avoiding RTOs.

Basically, RTO occurs in one of the following cases.

- The transmission window is too small to allow the generation of enough duplicate ACKs to trigger Fast Retransmit. This case is particularly critical at the beginning of the flow, since during the first-window transmission the RTO is very large (in most implementations it is set to 6 seconds).

0-7803-7605-6/02/$17.00

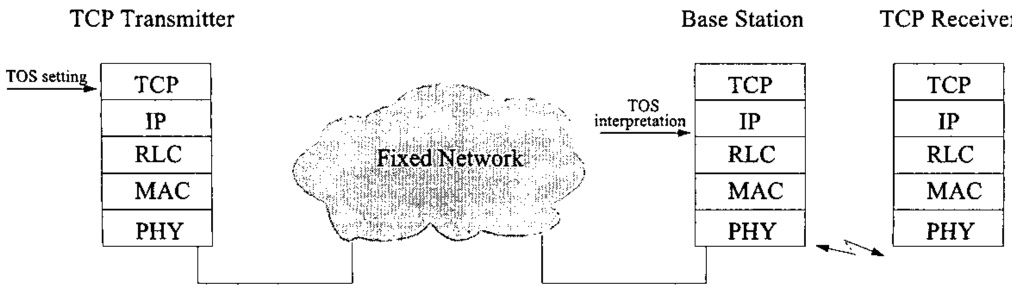

Fig. 1. System scenario.

- There are not enough duplicate ACKs to trigger the Fast Retransmit because less than 3 segments remain to be delivered to complete the file transfer.
- Most of the times in which the retransmission of a lost segment fails. In particular, consecutive RTOs translate into a remarkable penalty due to the backoff mechanism.

Accordingly, the proposed scheme protects most of the TCP segments whose lost is responsible of RTO:

- the first three segments of a flow
- the last three segments
- the first segment of a window which undergoes retransmission.

The protection of the first and last three segments of the flow results in a scheme which is particularly effective for short lived flows, as most of nowadays Internet traffic is. Different protection schemes can be chosen for different file sizes, as well as for the various versions of TCP. Also, we highlight that if we considered other QoS metrics as improvement objectives, different protection schemes would result to be more effective than the proposed one.

III. Implementation of the Proposed Protection Scheme

We assume that at the transport layer all TCP segments have the same size, which is equal to the Maximum Segment Size (MSS). At the link layer, TCP segments are fragmented into several link layer (LL) data units; and for the TCP segments which are identified as critical, some redundancy is introduced according to a FEC coding scheme. The transmission of LL data units over the radio channel is handled by the ARQ protocol. We assume that the ARQ protocol is based on the selective repeat technique. The receiver sends negative acknowledgments when erroneous or out-of-sequence data units arrive. The transmitter performs at most M_r attempts to deliver a LL data unit; after M_r failed transmissions a LL data unit is discarded together with the other data units belonging to the same TCP segment.

When FEC coding is applied, parity bits are added to information bits. We consider (n, k, t) BCH codes, where n are the bits in a codeword, k are the information bits, and t is the number of bits that can be corrected in each codeword. Being MSS the data segment size and L_{LL} the link layer data unit size expressed in bytes, each data segment is fragmented into N_f LL data units with $N_f = \lceil MSS \cdot n/(L_{LL} \cdot k) \rceil$. When no FEC is applied, the TCP segment is fragmented into $N = \lceil MSS/L_{LL} \rceil$ data units. A TCP segment is removed from the link layer buffer either when all the LL data units composing the segment have been successfully delivered to the receiver or any of the LL data units is discarded because the maximum number of transmission attempts has been reached. In the latter case, a whole TCP segment is lost.

The radio channel is modeled as a Gilbert channel [10], [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 are computed based on the fading process parameters and the length of the transmitted data unit. A bit is correctly received if the experienced signal-to-noise ratio (SNR) is above a certain threshold Γ, and is received in error otherwise [10], [11]. We assume that the noise power is constant and the average SNR is ΓF times larger than the noise power. F is called the *fading margin* and represents the maximum fading fluctuation that does not corrupt the transmission. Large values of F represent small error probabilities, while low values correspond to large error probabilities. The Gaussian correlation coefficient of two samples of the complex amplitude of a fading channel with Doppler frequency f_D, taken at time distance τ, is given by $\phi = J_0(2\pi f_D|\tau|)$, where $J_0(\cdot)$ is the Bessel function of the first kind and zeroth order. Therefore, the correlation properties of the fading process depend on the parameter $f_D|\tau|$ only, which is called *normalized Doppler frequency*. When $f_D|\tau|$ is small (< 0.1), the process is very correlated (slow fading) and long burst of consecutive errors occur, while for $f_D|\tau| > 0.2$ two samples of the process are almost independent (fast fading) [10], [11]. We assume that in state *good* the bit error rate, *p*, is equal to 10^{-10}, while in state *bad* *p* is equal to 10^{-5} [12]. When FEC is not considered, the error probability of a LL data unit is given by

$$Pe_{NO-FEC} = 1 - (1-p)^{8L_{LL}} \tag{1}$$

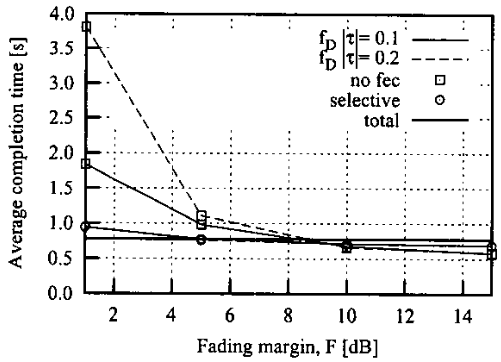

Fig. 2. Average completion time versus fading margin; $f_D|\tau| = 0.1, 0.2$, file size equal to 10 segments.

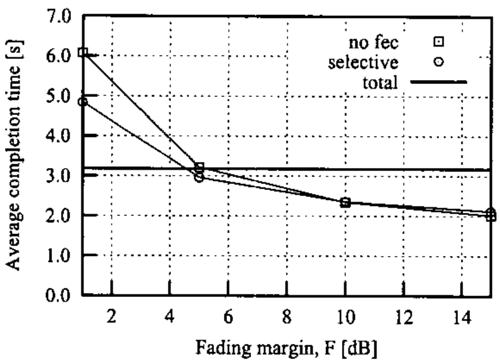

Fig. 3. Average completion time versus fading margin; $f_D|\tau| = 0.1$, file size equal to 50 segments.

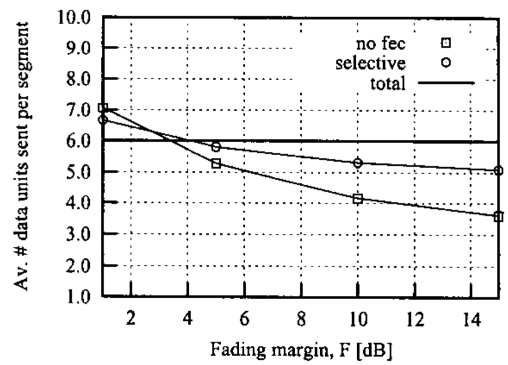

Fig. 4. Average number of data units sent per TCP segment; $f_D|\tau| = 0.1$, file size equal to 10 segments.

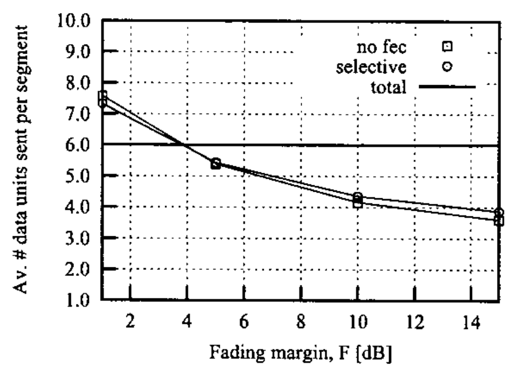

Fig. 5. Average number of data units sent per TCP segment; $f_D|\tau| = 0.1$, file size equal to 50 segments.

while, in the case of FEC [13], we have

$$Pe_{FEC} = 1 - \left[1 - \sum_{i=t+1}^{n} \binom{n}{i} p^i (1-p)^{n-i}\right]^{\lfloor \frac{8L_{LL}}{n} \rfloor} \quad (2)$$

where $\lfloor \frac{8L_{LL}}{n} \rfloor$ is the number of codewords in a LL data unit.

IV. Results

We derived simulation results by introducing some extensions to *ns-2*, [14]. The simulation setup comprises two links and three nodes: the TCP transmitter, the Base Station (BS), and the mobile station where the TCP receiver resides. The TCP transmitter and the BS are directly connected through a wired link which represents the fixed portion of the network (see Fig. 1). This link is assumed not to be congested, to have a 10 Mbps capacity, and to introduce a 30 ms one-way delay. The transmission rate over the wireless channel between the BS and the TCP receiver is equal to 384 Kbps, and the propagation delay is negligible. The transmission time of a LL data unit is equal to 10 ms [15]; the size of a LL data unit is equal to L_{LL} =480 bytes.

At the link layer, we assume a maximum number of transmission attempts equal to 6; this value showed to be a good trade-off between loss probability and average delay (curves are not shown due to the lack of space). We assume that acknowledgements are never lost on the reverse channel. A $(255, 131, 18)$ BCH code is used as FEC code.

Each simulation consists of a finite file transfer from the source to the mobile station using TCP NewReno at the transport layer. The TCP Maximum Segment Size (MSS) is set to 1460 bytes, and the number of LL data units per TCP segment is equal to N_f=6 when FEC is applied, and is equal to N=3 otherwise. The time granularity used by TCP (tic) is set to 0.5 s, the minimum RTO results to be equal to 1.5 s. The initial RTO is set to 6 s. Each point in the graphs is computed by averaging the results obtained by a number of simulation experiments of the order of tens of thousands.

The varying system parameters are the file size, which is equal to either 10 or 50 TCP segments, the fading margin, F, ranging from 1 to 15 dB, and the normalized Doppler frequency, namely $f_D|\tau| = 0.1, 0.2$. Performance is evaluated in three different cases: i) when no FEC is applied, this case is labeled by *no fec* in the plots; ii) when selective FEC coding is applied according to the proposed scheme, labeled by *selective* in the plots; iii) when FEC is applied to all TCP segments, labeled by *total* in the plots.

Fig. 2 shows the average completion time versus fading margin for file size equal to 10 segments, and $f_D|\tau| = 0.1, 0.2$. Small values of F correspond to large values of average error probability over the radio channel. Also, $f_D|\tau| = 0.1$ implies longer burst of errors than $f_D|\tau| = 0.2$, due to the higher correlation in the fading process. When F is small, using FEC leads to a great improvement in completion time, both in the case of *selective* as well as *total* packet protection. For large values of F, slightly smaller completion time is obtained when no FEC is applied; in fact, the code redundancy at the link layer makes the delay perceived by TCP segment increase. Observe that the completion time in the case of total protection is almost constant since error probability is very low. With selective protection, this performance metric tends to become insensitive to the radio channel conditions. The impact of $f_D|\tau|$ is only remarkable when no protection is applied and F is small. In fact, as it was shown in [16], in this case, given the error probability over the radio channel, longer bursts of errors translates into fewer loss events at the TCP layer.

Similar results are shown in Fig. 3 for file size equal to 50 segments and $f_D|\tau| = 0.1$. When F is small, the gap between the selective and total protection schemes is larger than in the previous case, however the improvement obtained through selective coding is still relevant. As F increases, selective coding gives almost the same performance as the *no fec* case which significantly reduces the completion time when compared to the total protection. Clearly, this behavior is due to the chosen protection scheme, that is particularly effective for short lived connections since coding is applied mainly to the first and last segments of the file.

As index of the bandwidth and energy waste, we consider the average number of LL data units transmitted per TCP segment. In Fig. 4 curves are presented as functions of the fading margin in the case of file size equal to 10 segments and $f_D|\tau| = 0.1$. When no FEC is applied, each TCP segment is fragmented into $N = 3$ LL data units. Due to error probability over the radio channel, data units may need to be retransmitted; thus, the average number of data unit transmissions per segment increases with the error probability, i.e., decreases with F. When FEC is applied to all segments, being $N = 6$ and the error probability very low, 6 LL data units per segment need to be transmitted. Therefore, for small values of F, better performance is obtained when FEC is applied to all packets. On the contrary, as channel conditions improve the *no fec* scheme is the most efficient solution. The selective scheme is a good trade-off between *no fec* and total protection as the fading margin varies. Again, when a file size of 50 segments is considered, the performance of the selective coding is closer to the one observed in the *no fec* case, as reported in Fig. 5.

V. Further Work

Additional experiments and further research are required in order to better understand the interaction between the transport level and the link layer, and in order to assess the effectiveness of our technique under different parameter settings. We are also planning to evaluate an alternative protection scheme in which FEC is applied at the link layer to all retransmitted data units irrespective of the TCP packet they belong to.

References

[1] A. De Simone, M.C. Chuah, and O.C. Yue, "Throughput performance of transport-layer protocols over wireless LANs," in *Proc. of Globecom'93*, December 1993.

[2] R. Caceres and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," *IEEE JSAC*, vol. 13, no. 5, pp. 850–57, June 1995.

[3] G. Xylomenos, G.C. Polyzos, P. Mähönen, and Mika Saaranen, "TCP performance issues over wireless links," *IEEE Communications Magazine*, pp. 52–58, April 2001.

[4] H. Balakrishnan, S. Seshan, and R.H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *ACM/Baltzer Wireless Networks Journal*, vol. 1, no. 4, December 1995.

[5] A. Bakre and B.R. Badrinath, "I-TCP: Indirect TCP for mobile hosts," in *Proc. of 15th International Conference on Distributed Computing Systems*, May 1995.

[6] K.-Y. Wang and S.K Tripathi, "Mobile-end transport protocol: an alternative to TCP/IP over wireless links," in *Proc. of INFOCOM*, 1998, pp. 1046–1053.

[7] K. Fall and S. Floyd, "Simulation-based comparison of Tahoe, Reno, and Sack TCP," *Computer Communications Review*, 1996.

[8] A. Chockalingam, M. Zorzi, and V. Tralli, "Wireless TCP performance with link layer FEC/ARQ," in *Proc. of ICC*, 1999, pp. 1212–1216.

[9] 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.

[10] 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.

[11] M. Zorzi and R.R. Rao, "Error control and energy consumption in communications for nomadic computing," *IEEE Transactions on Computers*, vol. 46, no. 3, pp. 279–89, March 1997.

[12] B.P. Crow, I. Widjaja, J.G. Kim, and P. Sakai, "Investigation of the IEEE 802.11 Medium Access Control (MAC) sublayer functions," in *Proc. INFOCOM'97*, Kobe, Japan, April 1997, pp. 126–33.

[13] J. G. Proakis, *Digital Communications*, 3rd ed. McGraw-Hill, Singapore, 1995.

[14] "ns-2, network simulator (ver.2), "http://www-mash.cs.berkeley.edu/ns.

[15] Third Generation Partnership Project (3GPP), "Technical specifications," March 2000, http://www.3gpp.org.

[16] M. Zorzi, A. Chockalingam, and R. R. Rao, "Throughput analysis of TCP on channels with memory," *IEEE JSAC*, vol. 18, no. 7, pp. 1289–1300, July 2000.