Analysis of TCP Latency over Wireless Links Supporting FEC/ARQ-SR for Error Recovery

Raja Abdelmoumen*
CRISTAL Laboratory, Tunisia
Email: Raja.Abdelmoumen@ensi.rnu.tn

Chadi Barakat
Projet Planète, INRIA-Sophia Antipolis, France
Email: Chadi.Barakat@sophia.inria.fr

Abstract— **We study in this paper the performance of TCP over a wireless link implementing hybrid FEC/ARQ-SR at the link layer. The study is done by simulating a large number of TCP transfers over a wireless link showing Bernoulli errors. We are motivated by how to tune link-level error recovery, e.g. amount of FEC, persistency of ARQ, so as to minimize the latency of TCP. We provide results for different physical characteristics of the wireless link (delay, error rate), different traffic loads and different file sizes. Our main finding is that the latency of TCP always improves with the persistency of ARQ, except for some extreme cases where the delay is large, files are small, and the loss rate is low. When adding FEC, the latency of TCP improves then deteriorates, and the deterioration is more pronounced in the case of large files, high loss rate and small delay. Another finding of our study is that with the hybrid mechanism, the wireless link is able to carry more traffic than when FEC and ARQ-SR are separately used.**

I. Introduction

The Transmission Control Protocol (TCP) considers the loss of a packet as an indication that the network is congested. This strategy for the detection of congestion results in a poor performance of the protocol when packets are lost in the network for other reasons than congestion [3], [5], [7], [10].Transmission errors on a wireless link form the main source for non-congestion losses. A TCP packet corrupted while crossing a wireless link is discarded before reaching the receiver, which results in an unnecessary window reduction at the TCP source. In the following, we call the corrupted TCP packets *non-congestion losses* or *link-level losses* since they appear at a level below IP.

Many solutions have been proposed in the literature to improve the performance of TCP when operating over paths with non-congestion losses [3], [5], [7]. Some of these solutions consist in enhancing TCP with additional mechanisms to help it to recover from non-congestion losses without reducing its window (explicit loss notification [5], loss predictors [8], etc.). Other solutions, e.g. I-TCP [4], propose to shield the sender from these undesirable losses by splitting the TCP connection at the entry of the wireless part (at the base station). Although they improve the overall performance, these solutions break the end-to-end semantics of TCP. To preserve the end-to-end semantics, other promising solutions propose to correct errors at the wireless link level by using a combination of FEC (Forward Error Correction) and ARQ (Automatic Repeat Request) [3], [5]. The drawback of FEC is that it consumes some extra bandwidth to transmit the redundant information. We showed in [1], [6] that there is an optimal amount of redundancy to add, above which the performance of TCP degrades instead of improving. ARQ does not consume much bandwidth, its drawback is that it increases the round-trip time (RTT), which may deteriorate the performance of TCP, if not done appropriately. The throughput of TCP is known to be inversely proportional to the average round-trip time [7]. Another problem of ARQ is the interference with TCP timeout. TCP retransmission timer may expire while the lost packet is being retransmitted over the wireless link. FEC does not cause neither an increase in RTT nor an interference with TCP timeout [3]. For these reasons, FEC-alone has been recommended to be used over long delay satellite links [3].

Clearly, there is a tradeoff in the tuning of link-level FEC and ARQ, and this tradeoff is dependent on the application type and the transport protocol used. For example, some applications as audio and video are sensitive to delay, so they prefer FEC over ARQ. Other applications such as bulk TCP transfers are more sensitive to throughput than packet delay, so they need a tuning that maximizes the channel throughput. Every application has some performance objective, and this objective, coupled with the dynamics of the transport protocol used by the application and the characteristics of the wireless link (loss rate, bandwidth, delay, etc.), allows to find the optimal tuning of the error recovery mechanism, i.e. the optimal amount of FEC, the optimal frame size for ARQ, the optimal maximum number of retransmissions, etc.

We study in this paper the performance of TCP over a wireless link combining FEC, ARQ-SR (ARQ Selective Repeat) and an in-order delivery of packets to IP. We are motivated by how to tune the parameters of a link-level hybrid FEC/ARQ-SR model so as to maximize the performance of TCP. We chose TCP since it is the transport protocol that carries more than 90% of Internet traffic [9]. ARQ-SR is an efficient ARQ scheme that avoids the unnecessary retransmissions that we might see with ARQ Go-Back-N. The main problem with ARQ-SR is that it desequences packets, hence a buffer is needed at the output of the wireless link for the purpose of resequencing packets and delivering them in-order to IP. Out-of-order packets are harmful to TCP since they result in duplicate ACKs, with the TCP source dividing its congestion window by two if three or more duplicate ACKs are received.

* This work was done while the author was doing his master internship within the Planète group at INRIA Sophia Antipolis.

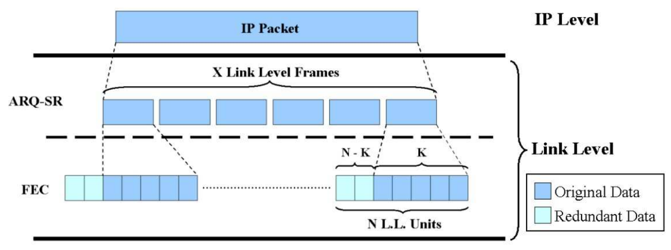

Fig. 1. The hybrid model FEC / ARQ-SR

Our study is done by simulating with `ns-2` [12] a large number of TCP transfers over a wireless link showing Bernoulli errors. For the purpose of the study, we implement a FEC/ARQ-SR error correction model with in-order delivery into `ns-2`. Then, we do our simulations for different physical characteristics of the wireless link (delay, error rate) and for different traffic loads and file sizes. For lack of space, we only present in this paper a summary of our results.

In a recent work [1], we considered the same problem of tuning FEC and ARQ-SR parameters, but in presence of TCP connections of infinite durations, i.e. TCP connections that last during the whole simulation time and that are feeded by infinite sources of data. For infinite-duration connections, the objective is simply to maximize the throughput of the aggregate TCP traffic. In our present paper, TCP connections are of finite durations; a connection is established at the beginning of the file transfer and is closed when the transfer ends. Hence, maximizing the throughput is no longer the main objective. Instead, the objective of the tuning should be to minimize the latency of TCP transfers under a certain traffic load. This objective can also be seen as maximizing the load that the wireless link can carry without impairing too much the latency of TCP. Considering TCP connections of infinite durations in [1] was a first step to understand the problem, since the model for the traffic was not very realistic. In this paper, we study the optimal tuning with a more realistic traffic model, where connections are of finite duration and where transferred files are of finite size.

Before presenting our model and the simulation results, we outline our main findings in [1] for the case of infinite-duration TCP transfers. As we will see, these findings almost hold for the case of finite-duration transfers. The first finding was that in all conditions, the throughput of the aggregate TCP traffic was continuously increasing with the number of retransmissions or equivalently with the persistency of ARQ-SR. The best tuning was then to always set the persistency of ARQ-SR to a large value, then to adapt the amount of FEC according to the load and to the wireless link conditions. When the loss rate of link-level frames was low (one frame lost from time to time), ARQ-SR alone was enough to maximize the throughput. We increased then the loss rate. The throughput of the wireless link started to drop, and it went to 0 with ARQ-SR alone. At moderate to high error rates, some amount of redundancy was needed to reach the maximum throughput. This amount was an increasing function of the error rate. Below and above the optimal amount of FEC, the throughput of TCP deteriorated, and this deterioration was more important when less FEC than the optimal was used.

We start this paper by a description of our model for FEC/ARQ-SR. In Section II we describe our model for TCP traffic and our performance objective. Then, we present the simulated scenario in Section IV. Section V summarizes our results, and Section VI concludes the paper and presents perspectives on our future research.

II. Model for FEC/ARQ-SR

This section describes our model for FEC/ARQ-SR that we implement in `ns-2` [12]. Consider a wireless link where data are transmitted in link-level (LL) frames. We denote by B the bandwidth of the link, and by D its one-way propagation delay. As we see in Fig. 1, each LL frame is divided into K link-level transmission units. An LL transmission unit can be a bit, a byte, or any other data unit. To the K units of an LL frame, we add $N-K$ redundant units that we obtain using a block code FEC as Reed-Solomon [10]. N is called the length of the code, K its dimension, and K/N its rate. We suppose that we have an erasure channel (the position of the error is known). An LL frame is decoded if we correctly receive K or more units of the frame at the output of the wireless link. A TCP packet that arrives at the input of the wireless link is divided into X frames. All TCP packets are of constant size P (MSS + TCP/IP header). Hence, $P = X \times K$ transmission units. To be delivered to the destination, the X frames of a TCP packet have to be well decoded. If FEC does not succeed to decode one LL frame, the link-level error recovery mechanism will resort to ARQ-SR for the retransmission of the frame. The retransmission will be done a maximum number of times that we call the persistency of ARQ-SR and that we denote by δ, $\delta = 0, 1, 2, \ldots$. $\delta = 0$ means that erroneous frames are not retransmitted. When δ trials are done and the frame does not get through the wireless link, ARQ-SR assumes that the frame cannot be locally recovered, and leaves for TCP the correction of the frame on end-to-end basis (by retransmitting the packet to which the frame belongs).

The ARQ-SR receiver at the output of the wireless link acknowledges each LL frame either with a positive ACK or a NACK. When a NACK is received at the input of the wireless link, the corresponding frame is directly retransmitted, and given priority over all other frames that have not yet been transmitted. One can imagine the use of a priority queue for retransmissions. The packets correctly received at the output of the wireless link are resequenced before being delivered to IP. An out-of-order delivery takes place only if a packet cannot be locally corrected, due to the failure of the retransmission of one of its frames or more. Note that packets are resequenced at the output of the wireless link on an aggregate basis, not on a flow basis. This simplifies the implementation of ARQ and respects the principle of layering in the Internet.

Concerning errors over the wireless link, we model them using a Bernoulli process, where transmission units are lost independently of each other with probability p. The error process is only applied to data units. ACKs of ARQ-SR and those of TCP are supposed to cross the wireless link without being lost given their small sizes. Usually, errors over wireless links are modeled by a Markov chain of two or more states since they appear in bursts [6], [10], [11]. We choose the Bernoulli model for the simplicity of the analysis it allows. The Bernoulli model is known to hold over fast fading channels (high speed mobile users) [11] and when interleaving is used [5]. We are only interested in this paper by the impact of the average error rate. The impact of burstiness of errors will be the subject of a future research.

III. Model for TCP traffic

We consider files that arrive as a Poisson process of constant rate λ, and that are transmitted using TCP over the wireless link. All files are of same size S and are transmitted over Internet paths having the same characteristics. When a file arrives, a TCP connection is established to transmit it. The connection is released when the file transfer ends. For each transfer that ends before the end of the simulation, we measure its latency, then we average over all latencies. This average latency is the performance measure we will consider in the following to evaluate a certain tuning of FEC/ARQ-SR. It is a function of FEC, ARQ-SR, the characteristics of the wireless link and the traffic load $\lambda \cdot S$. We measure the latency of a TCP transfer as the time elapsed between the transmission of the first packet of the file and the reception of the ACK that acknowledges the receipt of the whole file.

To avoid the congestion of the wireless link, we always make sure that the average load $\lambda \cdot S$ is less than the available bandwidth. The available bandwidth is equal to B decremented by a factor α that accounts for the amount of bandwidth wasted on FEC and ARQ-SR retransmissions. αB is the maximum rate of ”packets” we can get at the output of the wireless link for a certain error rate p and a certain tuning of FEC and ARQ-SR. When the condition of stability $\lambda \cdot S < \alpha B$ is satisfied, we are sure that the TCP transfers are able to end and that they will not be aborted due to lack of bandwidth. In [2], we found that under our assumption on the error process, α is equal to:

$$\alpha = (1 - P_F)^{X-1}(1 - P_T)\frac{K}{N},$$

where P_T is the probability that a frame is corrupted,

$$P_T = \sum_{i=N-K+1}^{N} \binom{N}{i} p^i (1-p)^{N-i},$$

and P_F is the probability to abort the transmission of a frame due to the loss of the original transmission and the δ retransmissions, i.e. $P_F = P_T^{\delta+1}$. Later, we only consider scenarios that satisfy the above condition of stability.

IV. Simulation scenario

Our model for FEC/ARQ-SR is only applied to the wireless link, and is transparent to the rest of the network and to TCP peers. With no loss of generality, we set K to 10, X to 6 and the size of link-level units to 25 bytes. TCP packets are then of constant size equal to 1500 bytes (MTU of Ethernet).

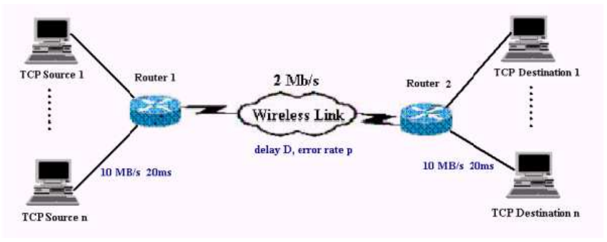

Fig. 2. Simulated network topology

The values given to K and X are randomly chosen. Our purpose is not to optimize X and K, but rather to optimize the amount of FEC to be used and the persistency level δ, given the values of K, X, and of the other parameters of the model. Nevertheless, the optimization of K and X is an interesting problem given the tradeoff it involves. We leave this optimization for a future research. For instance, for constant packet size P, the size of LL frames decreases when X increases, which decreases the probability that an LL frame is corrupted and the bandwidth wasted on retransmissions. This should improve the performance of ARQ. However when we increase X, K decreases as well as the number of redundant units per LL frame, which may deteriorate the performance since frames can now resist to less errors. The optimization problem becomes more interesting when we allow P to change since P decides on the rate with which TCP increases its congestion window [6], [13].

We consider several scenarios that are derived from the network topology shown in Figure 2. Several TCP sources generate files of constant size S according to a Poisson arrival of total rate λ. The sources use the NewReno version of TCP and enable the Delay ACK functionality [13]. Sources and destinations are connected to the wireless link via 10 Mbps links having a one-way propagation delay equal to 20 ms. The bandwidth of the wireless link B is set to 2 Mbps, its delay D varies between 20 ms and 200 ms depending on the scenario. The values we give to D model different types of wireless links ranging from terrestrial links to satellite ones. Transmission units are lost (corrupted) over the wireless link with probability p ranging from 10^{-3} to 10^{-1}. For transmission units of 25 bytes, the Bit Error Ratio is approximately equal to $p/(25 * 8)$. We vary the traffic load $\rho = \lambda \cdot S/B$ between 25% and 75% of the link bandwidth B and we set the file size S between 10 KB and 1 MB.

To avoid the impact on TCP latency of any other factor than errors on the wireless link, we take the following measures: (i) The wired links are completely reliable. (ii) The size of TCP advertised window is very large, up to 2000 packets. (iii) The buffers in network routers are very large, they can store up to 500 packets. (iv) The buffer used for resequencing packets at the output of the wireless link is very large. Under these conditions, it is clear that the wireless link is the only bottleneck on the path of the TCP connections and that congestion losses do not appear when the condition of stability is verified. We want to optimize the parameters of FEC and ARQ-SR in this best case setting. The optimization problem is not very meaningful when the TCP connections are constrained by some other parts of the network.

V. Simulation results

We begin by studying separately the effect of FEC and ARQ-SR on TCP latency. Second, we study the performance of the hybrid model, i.e. FEC and ARQ-SR combined together. We generate more than 500 file transfers per simulation and per scenario[1], then we compute the average TCP latency. For each scenario, we repeat the simulation 10 times with different seeds. By doing that, confidence intervals are of very small values. For lack of space and given the large number of scenarios, we only present the most significant results.

A. FEC alone

The case FEC alone can be obtained by setting $\delta = 0$. In this case, LL frames are not retransmitted but only protected by FEC. The most challenging scenario is when the error rate is high, p =0.1, and when the delay is large, D =200ms. We first look whether we can transfer a traffic that accounts for 75% of the link bandwidth. For this load, we notice that all the values of N do not respect the stability condition. Therefore, for a high error rate, $p = 0.1$, FEC alone require lot of bandwidth to correct all errors, which limits the traffic we can transmit over the wireless link (at this error rate, more than 25% of bandwidth need to be reserved for the redundant information).

Fig. 3 shows TCP latency in logarithmic scale as a function of the parameter N of FEC. Three lines are plotted which correspond to three distinct values of delay, D = 20, 100, 200 ms. In all three cases, p is set to a high value 0.1, ρ is set to 50% and S to 100KB. The values of N which are not considered do not satisfy the stability condition. By increasing the amount of FEC, the latency decreases until some optimal value of N then increases. The same behavior has been observed in [1], [6]. We notice that more FEC is required when the delay D increases. This is because at high delay, TCP is more sensitive to errors and needs more protection. We also notice in the figure that the deterioration of the latency on the right-hand side of the optimal point is more important when the delay is small. In fact, the addition of FEC increases the transmission and queuing time of TCP packets, which increases the end-to-end delay. This increase in the end-to-end delay is relatively less important when the link delay is large, which leads to this smaller negative impact of FEC on the right-hand side of the optimal point.

B. ARQ-SR alone

The case ARQ-SR alone is obtained by fixing the value of N to 10 ($N = K = 10$). Thus, LL frames are transmitted without being protected by FEC. When they are lost, the frames are retransmitted a maximum number of times equal to δ.

Our first observation is that for long transfers, e.g. S =1MB, the latency of TCP improves when δ increases for all chosen values of p, D and ρ . This monotonous improvement has been already observed in [1]. Next, we show that this is not always the case with short transfers.

[1]Scenarios differ by the values of D, p, λ, S, δ, and N.

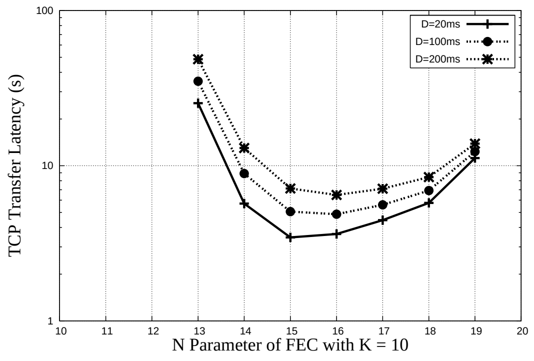

Fig. 3. FEC alone, $\rho = 50\%$, S = 100KB and $p = 0.1$

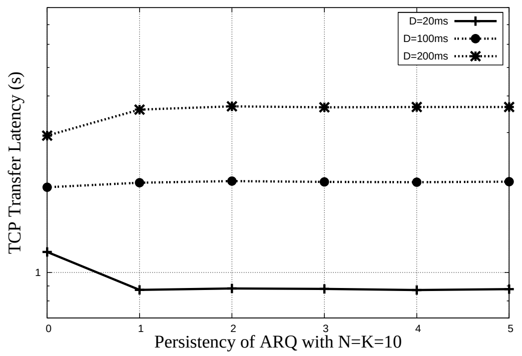

Fig. 4. ARQ-SR alone, $\rho = 75\%$, S = 10KB and $p = 0.001$

Let us first examine the effect of ARQ-SR in the challenging scenario of high error rate, p =0.1. For the values of the traffic load ρ =75% and 50%, all values of δ do not respect the stability condition. Comparing this results with that obtained by FEC alone, we realize that FEC is more interesting than ARQ-SR when the error rate is high, since with FEC alone we are able to reach a throughput equal to 50% of the link bandwidth. To obtain higher throughput with ARQ-SR, one has to reduce the frame size K, or add some units of FEC in order to reduce the frame error rate.

Second, we consider the case of short transfers S =10KB, small loss rate p =0.001, high traffic load ρ =75% and we vary the link delay D. In Fig. 4, the TCP latency is plotted as a function of the persistency δ. We notice that ARQ-SR improves monotonously the latency when the delay is small D =20ms, however when the delay is large, the latency of TCP increases with δ.

We explain this observation by the fact that the RTT of TCP undergoes large and unfrequent (error rate is low) variations introduced by retransmissions when the delay is large and the error rate is low, which causes the expiration of the retransmission timer and the reset of the congestion window to 1 segment. To better understand this increase in the latency for large delays, we simulate the same scenario in Fig. 4 but this time after increasing the minimum timeout value of TCP to 10 secondes. With such large value for the minimum timeout, we are sure not to have any false timeout caused by frames that are being retransmitted at the link level. The

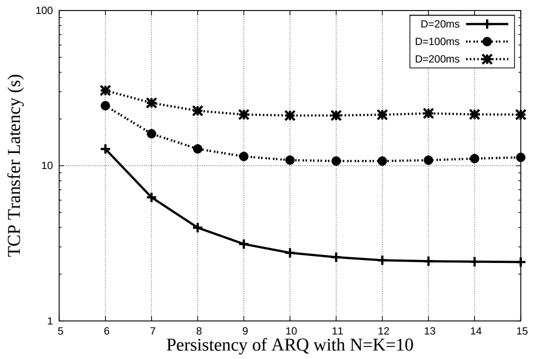

Fig. 5. ARQ-SR alone, $\rho = 25\%$, $S = 10$KB and $p = 0.1$

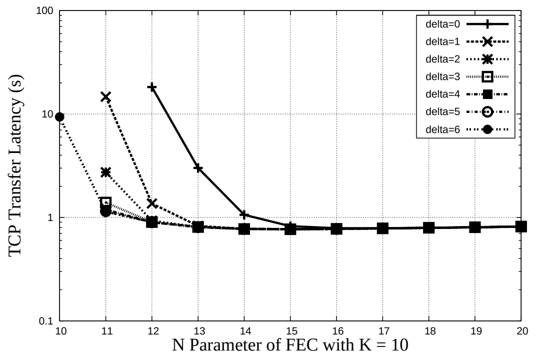

Fig. 6. Hybrid model, $\rho = 25\%$, $S = 10KB$, $p = 0.1$ and $D = 20$ms

result of the simulation shows that the latency decreases now with δ for long link delay. We can then conclude that the increase in the latency has been caused by some false timeouts that appear when the RTT suddenly increases due to a frame retransmission at the link level.

Now, we study the same case seen in Fig. 4 but with a high loss rate p =0.1 (and the minimum value of the timeout set to its standard value in `ns-2`). The traffic load is equal to 25% of link bandwidth. The results are plotted in Fig. 5. Surprisingly, the latency is decreasing with δ, even though the delay is large. Indeed, when the error rate is high, most of the frames are lost and retransmitted at the link level, so TCP retransmission timer ends up by adapting quickly its value to the maximum RTT caused by retransmissions.

C. Hybrid FEC/ARQ-SR

Consider the hybrid FEC/ARQ-SR scheme. We first focus on the challenging scenario where p =0.1, ρ =75% and D =200ms. By combining FEC and ARQ-SR, the hybrid model manages to transfer a traffic load of 75% of the link bandwidth for a single amount of FEC ($N = 13$) and a number of retransmissions $\delta > 0$. Neither FEC alone nor ARQ-SR alone were able to support such a load.

We consider next the scenario represented in Fig. 6 where ρ =25%, p =0.1, S =10KB and D=20ms. We see different lines in the figure that correspond to different values of the persistency. The line $\delta = 0$ shows the effect of FEC alone on the TCP latency. The values of the TCP latency on the y-axis, i.e. for $N = K = 10$, shows the effect of ARQ-SR alone. We remark in this figure that FEC alone is able to give a very good performance, but a large amount of FEC is needed. By enabling ARQ-SR and setting δ to a large value, a better performance can be obtained with a smaller amount of FEC; ARQ-SR retransmissions help FEC to correct errors when the error rate is high. The same result is observed at high error rate when we varying the other parameters of the model: the best latency is always obtained by adjusting the amount of FEC and setting δ to a large value.

In the other scenarios when the delay and the error rate are small, our results show that ARQ-SR alone gives a good performance close to the optimal one. On contrary, for large delay and small error rate, our results show that it is appropriate to use FEC alone with a small amount of redundancy.

VI. Conclusions and perspectives

The main finding of our analysis is that for finite TCP transfers and Bernoulli errors, the use of ARQ-SR always improves the latency of TCP, except for some extreme cases where the delay is large, files are small, and the loss rate is low. The addition of FEC improves first the latency of TCP then deteriorates it, so there is always an optimal amount of FEC to add. By combining FEC and ARQ-SR, our study shows that the wireless link is able to carry more traffic than when FEC and ARQ-SR are used alone. Also, the hybrid model achieves the best performance at high error by setting the persistency to a large value and by tuning optimally the amount of FEC.

Our work can be extended by studying analytically the latency of TCP over a hybrid FEC/ARQ-SR wireless link. We also want to check whether our findings hold in the case of bursty transmission errors.

References

[1] A. Al Fawal, C. Barakat, ”Simulation-based study of link-level hybrid FEC/ARQ-SR for wireless links and long-lived TCP traffic”, *WiOpt’03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Mar. 2003.

[2] C. Barakat, A. Al Fawal, ” Analysis Of Link-Level Hybrid FEC/ARQ-SR For Wireless Links and Long-Lived TCP traffic”, *INRIA Research Report RR-4752*, Feb. 2003

[3] M. Allman, D. Glover, L. Sanchez, ”Enhancing TCP Over Satellite Channels using Standard Mechanisms”, *RFC 2488*, Jan. 1999.

[4] A. Bakre, B. R. Badrinath, ”I-TCP: Indirect TCP for Mobile Hosts”, *International Conference on Distributed Computing Systems (ICDCS)*, May 1995.

[5] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, R. Katz, “A comparison of Mechanisms for Improving TCP Performance over Wireless Links”, *ACM SIGCOMM*, Aug. 1996.

[6] C. Barakat, E. Altman, ”Bandwidth tradeoff between TCP and link-level FEC”, *Computer Networks*, vol. 39, no. 2, pp. 133-150, Jun. 2002.

[7] C. Barakat, E. Altman, W. Dabbous, ”On TCP Performance in a Heterogeneous Network : A Survey”, *IEEE Communications Magazine*, vol. 38, no. 1, pp. 40-46, Jan. 2000.

[8] D. Barman, I. Matta, ”Effectiveness of Loss Labeling in Improving TCP Performance in Wired/Wireless Networks”, *ICNP*, Nov. 2002.

[9] CAIDA: The Cooperative Association for Internet Data Analysis, http://www.caida.org

[10] A. Chockalingam, M. Zorzi, R.R. Rao, ”Performance of TCP on Wireless Fading Links with Memory”, *IEEE ICC*, Jun. 1998.

[11] A. Kumar, J. Holtzman, ”Performance Analysis of Versions of TCP in a Local Network with a Mobile Radio Link”, *Sadhana: Indian Academy of Sciences Proceedings in Engg. Sciences*, Feb. 1998.

[12] The LBNL Network Simulator, `ns-2`, http://www.isi.edu/nsnam/ns/

[13] W. Stevens, ”TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms”, *RFC 2001*, Jan. 1997.