TCP Performancance Improvement over HF Channels by Lower-Layer Parameter Tuning

Márcio Barroso Toscano Dantas, Juraci Ferreira Galdino & Ernesto Leite Pinto

***Abstract*— The advent of new High-Frequency (HF) transmission technologies has highlighted the necessity of researches focused on the TCP performance over HF links. Some recent investigations on TCP performance improvement over other wireless communications systems have shown that the use of Automatic-Repeat-Request Selective-Repeat (ARQ-SR) protocols at data link level may provide significant advantages. The main objective of the present work is to evaluate by computer simulation the TCP performance over HF channels, and the improvements that can be obtained by employing an ARQ-SR protocol and/or a Forward-Error-Correction (FEC) coding strategy in the link layer. Besides, the influence of the ARQ-SR parameters on the TCP performance is also addressed.**

***Index Terms*— ARQ-SR Protocol, FEC, HF channel, TCP throughput.**

I. Introduction

It is largely recognized that HF subnets provide long distance links without repeaters and may be a lower-cost and less vulnerable alternative to satellite networks. They are particularly interesting for the following scenarios: military operations; data communications to remote locations where installation of conventional networks is very hard; in areas affected by accidents, strikes or natural disasters, and communications with aircrafts, ships and boats.

Despite these advantages, HF communication systems typically have low transmission rates and high Bit Error Rates (BER), due to their limited bandwidths (typically 3 KHz) and to multipath fading effects.

With the emergence of new High Frequency (HF) communication systems [1] potentially feasible for integration with the conventional networks based on Transmission Control Protocol / Internet Protocol Architecture (TCP/IP), the HF channel has become the main object of new researches focused on performance improvement of this integration.

The great interest in the use of the TCP/IP stack as a form of internetworking between heterogeneous networks is justified by its well-established standardization and use[1], in conjunction with the popularization of Internet. Besides, there has been a wide variety of available applications (HTTP, SMTP, FTP, databases, etc.) for many corporate, domestic, academic, government and military environments, which stimulates the convergence of applications, infrastructure and management networks.

However, several studies indicate that the TCP protocol, originally designed for wired networks, produces low throughput in wireless networks. Besides, TCP/IP-based networks present very poor performance over typical HF channels [12].

A significant research effort has been invested in developing strategies to improve TCP performance in wireless networks [2], [3], without modifying the TCP end-to-end semantics. These investigations can be split into two main classes. One of them try to modify TCP state machine to improve TCP compatibility to wireless network. The other one tries to use the data link layer protocols and physical-layer countermeasures, as a means to provide more reliability to the transport layer in these scenarios.

The use of the ARQ-SR protocols and FEC in the data link layer has been shown to provide significant advantages for TCP performance improvement in the context of other wireless communications networks, as shown in [4], for instance.

Following this trend, in the present work we investigate the impact of ARQ-SR protocol, FEC and ARQ-SR/FEC association in the performance of TCP over HF links. Besides, we also investigate the tuning of ARQ-SR parameters as a tool to improve TCP performance in this context.

The rest of the paper is structured as follows. The next section describes the TCP flow, errors and congestion control mechanisms in heterogeneous networks. Section III discusses the causes of TCP throughput degradation in wireless channel. Data link layer strategies for TCP performance improvement in this kind of channel are presented in Section IV. The main architectures of data-link protocols for HF communications are presented in section V. Section VI describes the models and tools here used for performance evaluation by simulation. Several results are presented in section VII. Finally, concluding remarks are given in Section VIII.

II. TCP flow, errors and congestion control mechanisms in heterogeneous networks

The TCP protocol is intended to provide a reliable service for data transport over a network with disturbances that can lead to errors during the transmission of information.

A large network, such as the Internet, has high complexity, considering that each of its various sub-networks presents different topologies, bandwidths, delays, packet sizes, numbers of users and other parameters. Thus, TCP tries to dynamically adapt to changes in network parameters as well as to answer or resolve failures that arise during transmission. This is done through the TCP mechanisms of flow control, error control

Manuscrito recebido em 30 de novembro de 2010; revisado em 1 de abril de 2011.

M. B. T. Dantas (mtoscano75@gmail.com), J. F. Galdino (galdino@ime.eb.br) e Ernesto L. P. (ernesto@ime.eb.br) pertencem ao Instituto Militar de Engenharia - IME. Pça. General Tibúrcio, 80 - Urca - Rio de Janeiro - RJ - Brasil - 22290-270.

[1]TCP is the transport protocol that carries more than 90% of Internet traffic [13].

and congestion control, that are discussed in the following subsections.

A. TCP flow control mechanisms in heterogeneous networks

The TCP protocol incorporates flow control mechanisms to ensure communications between terminals with different transmission rates, since in these cases the sender should send TCP segments [2] at a rate compatible with the capacity of the receiver.

The TCP flow control is accomplished by manipulating two parameters: the congestion window size (*cwnd*) and the TCP receiver window size, also named advertisement window size (*awnd*).

The transmission window size corresponds to the amount of segments that can be buffered at the transmission terminal, i. e., it is the maximum number of unacknowledged segments that can be dealt with in the transmission side.

The transmission window size is set as the minimum between *cwnd* and *awnd*. Most current receivers have large receive buffers, i.e., large values of *awnd*, so the transmission window size effectively defined by the *cwnd* parameter.

The adaptation of the congestion window is performed by the Slow-Start Algorithm, that can be summarized in the following:

1) At the beginning of a connection or after the *timeout*, make $cwnd = 1$;
2) For every received ACK, make $cwnd = cwnd + 1$.

The effect of this mechanisms is the exponential increase of *cwnd*, rapidly leading to a *timeout* occurrence.

After the timeout event caused by Slow Start, the Congestion Avoidance Algorithm is triggered, with the aim of producing a more conservative increase of *cwnd*. This algorithms consists of the following:

- For every ACK received $\Longrightarrow cwnd = cwnd + \frac{1}{cwnd}$.

To ensure a sending rate compatible with the capacity of the network, flow control and congestion control algorithms are implemented jointly as will be discussed in the following subsections.

B. TCP error control mechanisms

TCP error control mechanisms uses two sources of segment-loss signalling: the receipt of three duplicate ACKs and the occurrence of retransmission *timeout* period (RTO) expiration.

The TCP acknowledgment process uses the sequence numbers contained in the TCP header to identify the segments associated to TCP end-to-end connection between transport layers. This process is cumulative, so the ACK segment informs to the sender the number of the next segment (sequence number) which is expected to be received, here denoted by n. This indicates that all the segments with sequence number equal less than n have been received correctly.

Then, when the sequence number n is expected and a higher number is received, a new ACK is sent requesting the n-th segment. This is called duplicate ACK. Therefore, arrivals of out of order segments generate duplicate ACKs to the sender. In fact, the reception of out-of-order segments, and the corresponding generation of duplicate ACKs are normal in networks with dynamic routing, where packets travel through different routes, with different delays. Under abnormal conditions another reason for receiving out of order segment arises, which is segment loss.

The receipt of three duplicate ACKs by the sender is regarded as an indication that a segment loss occurred, being the inferred lost segment immediately retransmitted. This algorithm is called Fast Retransmit. The detection of segment loss through duplicate ACKs suggests that the network is under a moderate congestion condition.

The purpose of the Fast Retransmit Algorithm is to make the segment retransmission process more efficient. It intends to avoid long waits for *timeout* expiration, and this way speed up the retransmission.

The segment loss detection by *timeout* suggests the occurrence of burst losses due to severe congestion or due to wireless propagation conditions. In other words, the occurrence of bad scenarios for the packet traffic flowing.

The value of RTO is a crucial parameter for TCP performance in wireless channels, since it directly affects the speed of packet-loss detection.

Due to the high variability of network conditions, it is a very difficult task to precisely infer about the instants of positive acknowledgments receptions. As an attempt to deal with the variations of delay in the Internet, TCP uses the Adaptive Retransmission Algorithm that aims to establish suitable values for *timeout* timer by continuously monitoring the variable Round Trip Time (RTT) of each connection, which is the time between the segment transmission and the reception of the corresponding positive acknowledgment.

Denoting by RTT_p the previous estimate of the mean RTT and by RTS (round trip sample) the current RTT measurement, the new estimate of the mean RTT (RTT_c) is calculated as follows:

$$Error = RTS - RTT_p, \tag{1}$$

$$RTT_c = RTT_p + \gamma \times Error, \tag{2}$$

where $Error$ is the difference between the current RTT sample and its previously estimated mean, and γ is a factor between 0 and 1 typically set to $1/8$.

The *timeout* value is updated according to the equations below:

$$DEV_c = DEV_p + \rho \times (|Error| - DEV_p), \tag{3}$$

$$\textit{Timeout} = RTT_c + \eta \times DEV_c, \tag{4}$$

where DEV_c and DEV_p denotes, respectively, the current and previous estimates of the average fluctuation of the difference $Error$. On the other hand, ρ is a factor between 0 and 1 (typical value ρ = $1/4$), which weights the effect of a new sample in the average fluctuation, and η controls the effect of this average fluctuation in the *timeout* updating. A typical value for η is = 4 [16].

[2]The term segment is here used to refer specifically to the protocol data unit of the transport layer

Therefore, the value of the *timeout* timer is adaptive. This is an important feature for the use of TCP over high delay channels, such as HF.

The Karn's algorithm states that in case of a segment-loss detection, the *timeout* timer is doubled.

C. TCP congestion control mechanisms in heterogeneous networks

The TCP protocol uses the algorithms discussed in the previous subsections and the Fast Recovery Algorithm to perform congestion control.

In severe congestion situation (*timeout* event), the Slow Start and Congestion Avoidance Algorithms are triggered jointly under the control of a variable called ssthresh (Slow Start threshold) as follows:

1) At the beginning of connection, set $cwnd = 1$ and $ssthresh = 0$: *Slow Start*;
2) *timeout* event $\Longrightarrow ssthresh = \frac{cwnd}{2}$;
3) $cwnd = 1$; restart *Slow Start*;
4) If $cwnd < ssthresh$, *cwnd* is manipulated by *Slow Start*;
5) If cwnd $\geq$ ssthresh, *cwnd* is manipulated by *Congestion Avoidance*.

On the other hand, in case of moderate congestion the Fast Recovery Algorithm is triggered in conjunction with Fast Retransmit, in order to manipulate *cwnd*. The description of the algorithms is summarized below:

1) Upon receiving 3 consecutive duplicate ACKs or the occurrence of a *timeout* event, make $ssthresh = \frac{cwnd}{2}$;
2) Retransmit the presumed lost segment: *Fast Retransmit*;
3) $cwnd = ssthresh + 3$;
4) Each duplicate ACK received $\Longrightarrow cwnd = cwnd + 1$;
5) Upon receiving a new ACK of retransmitted segment, make $cwnd = ssthresh$.

III. Low TCP Throughput in wireless channel

As discussed in the previous Section, the TCP protocol has been conceived under the assumption that all segment losses are caused by congestion in the network.

This assumption is valid in wired networks environments, which generally produce low BER. In wireless channels, however, the BER is typically higher, so segment losses due to channel errors are mistakenly regarded by TCP as indications of network congestion.

Hence, to assist in alleviating the congestion and to avoid successive segment losses, the TCP congestion control mechanism activates the above mentioned procedures. They reduce the *cwnd* size in order to reduce the transmission rate (assuming that *cwnd* is smaller than *awnd*).

In addition, after each *timeout* event, TCP duplicates the value of RTO. So, if the channel undergoes a fading event of long duration, several consecutive retransmission failures may occur and lead to a long period of inactivity of the TCP connection.

These mechanisms jointly reduce the efficiency of transmission. Furthermore, in HF channels this effect is stronger, since these channels are characterized not only by high error rates, but also by the occurrence of burst errors and by having long transmission delays.

Another fundamental aspect in the analysis of TCP over wireless channels is the presence of errors in the return channel. Because of the relatively small size of the ACK segments, they can also experience losses induced by congestion and/or transmission errors. In this context, the TCP acknowledgment mechanism based on ACK feedbacks could be affected and even dominated by poor conditions on the reverse channel. As a consequence, a sender could mistakenly interpret a bad return channel condition as congestion on the forward channel and unnecessarily reduce the sending rate, leading to TCP throughput degradation.

IV. Data link layer strategies for TCP throughput improvement over wireless channel

Some frequently applied strategies for improving TCP throughput in wireless environments involve the use of data link layer protocols and FEC (*"Forward Error Correction"*) techniques with the objective of hiding segment losses from upper layers.

With respect to the data-link layer, the most used technique has been ARQ-SR. It performs retransmissions, fragmentation and reassembly, and thereby requires buffering both in the transmitter and in the receiver. Furthermore, the receiver has a *timeout* control for each sent frame (protocol data unit of the data-link layer).

When this technique is applied in association with TCP/IP, each IP packet is fragmented into smaller frames and the transmitter sends several frames per forward transmission. If a *timeout* event occurs, the receiver sends a NACK (*"Negative Acknowledgment"*) frame to the transmitter, in order to inform that a data frame has been unsuccessfully received. Immediately after receiving a NACK frame, the transmitter resends the indicated information frame. This procedure is repeated until the frame is successfully delivered or the number of retransmissions exceeds the value of the persistency parameter of the ARQ-SR protocol, denoted by δ. In the latter case, the transport protocol resends an entire TCP segment. After receiving all frames successfully, the receiver reassembles the original IP packet.

The disadvantage of using ARQ-SR is the large variation delay due to the accumulation of retransmissions of data-link and TCP protocols, caused by the use of incompatible timers [8]. In fact, for TCP to perform well in conjunction with an ARQ-SR protocol, the *timeout* value of the former has to be greater than the one of the later (i. e., $timeout_{TCP} > timeout_{ARQ}$).

Another problem with ARQ-SR is that it desequences frames, hence a buffer is needed at the output of the wireless link for the purpose of resequencing frames and delivering them in order to IP layer.

FEC techniques, used alone or in association with an ARQ-SR technique, may alleviate or solve this problem, since they do not rely on retransmissions. The drawback of FEC is that it may consume significant extra bandwidth in pro-actively transmitting redundant information.

The effect of these drawbacks must be evaluated considering the Quality of Service (QoS) requirements of each application type. For instance, [14] shows that for applications that are sensitive to delay it could be preferable not to use an ARQ mechanism and rely only on the use of FEC instead. We stress that our work focuses on the evaluation of these link layer strategies in HF channels aiming applications that are insensitive to delay, such as File Transport Protocol (FTP).

V. The main architectures of data-link protocols for HF communications

Current HF communication systems equipped with interface for TCP/IP networks adopt two main architectures for the data link layer protocols: the one of military standards (STANAG or MIL-STD) and that of AX.25 protocol.

The STANAG 5066 (*"Profile for HF Radio Data Communications"*) [11] is a NATO specification that describes an interface for data applications to share an HF modem. It addresses data format, ARQ protocol and procedures required for interoperable data communications over HF links.

This profile includes a mechanism for adapting the data rate in accordance with the prevailing channel conditions and an ARQ mechanism whose parameters are summarized in Table I:

TABLE I
PARAMETERS OF ARQ PROTOCOL - STANAG 5066.

PARAMETERS	TYPE SIZE
ARQ Type	Selective Repeat
Frame size	220 bytes
Frame Header Size	20 bytes

On the other hand, the U.S. military standard MIL-STD-188-110B [11] specifies that the error correction coding for modems operating at fixed frequency should comply with Table V.

DATA RATE (bps)	EFFECTIVE CODE RATE	METHOD FOR ACHIEVING THE CODE RATE
4800	(no coding)	(no coding)
2400	1/2	Rate 1/2
1200	1/2	Rate 1/2 code
600	1/2	Rate 1/2 code
300	1/4	Rate 1/4 code repeated 2 times
150	1/4	Rate 1/8 code repeated 4 times
75	1/2	Rate 1/2

Another group of data link layer protocols for HF communications follow the AX.25 specifications widely used by the amateur radio community, that establish some default settings of the ARQ protocol, as presented in Table V:

PARAMETERS	TYPE SIZE
ARQ Type	Selective Repeat
Frame size	256 bytes
Frame Header Size	20 bytes

The AX.25 protocol uses a frame address field of 14 bytes, remaining 6 bytes for control information. The size of the frame header is an important parameter for evaluating the system efficiency, since it is intrinsically linked to the overhead on each frame.

In this work, the performance of TCP over HF channels is evaluated under different configurations of the above mentioned parameters, beside of persistency parameter.

VI. Models and Tools for Performance Evaluation

A. Simulation Environment

We used the Network Simulator - ns-2 [5] to simulate a TCP transfer and FEC/ARQ-SR error correction model with in-order delivery of frames to IP protocol. In relation to the bit error model at the physical layer, a Hidden Markov Model (HMM) was used.

Fig. 1 illustrates the communication scenario considered in this work. It is composed of a FTP server connected to a LAN, a terminal connected to another LAN and two HF radios equipped with TCP/IP (HF Gateway), that connect the two LANs.

The topology illustrated in Fig. 1 was translated and loaded into the ns-2, giving origin to the simulation model presented in Fig. 2.

In all simulations the steps of authentication and access to the FTP server were considered to be concluded. Thus, the performance evaluation was based on FTP file transfer from server to terminal.

We assumed that the HF radios were equipped with modems based on the MIL-STD-188-100B recommendation, and performed uncoded transmission at a rate of 4800 bps, with BER of 10^{-3}. When using FEC (convolutional coding), the BER was supposed to be 10^{-5}. The assumed coding rates and the corresponding user bit rates are shown in Table VI-A.

DATA RATE (bps)	FEC Rate	BER
4800	1	10^{-3}
2400	1/2	10^{-5}

Besides, it was assumed that the distance between terminals was 2900 km and that the propagation mechanism was total reflection in the ionospheric F2 layer, whose height is approximately 380 km. Assuming free space propagation, the corresponding propagation delay was approximately 10 ms.

It should be noticed that when the propagation delay is Δ_P the total delay may be expressed as:

$$\Delta_T = \frac{N}{R} + \Delta_P, \tag{5}$$

where R denotes the transmission rate and N denotes the packet/frame size.

We also assume that the reverse channel is completely reliable, i.e., transmission error occurs only in the forward channel, as illustrated in Fig. 2, since the size of the IP packet of the return channel is much smaller then the one of the forward channel. This is a reasonable assumption that has been adopted in several works focused on the performance evaluation of communication protocols over wireless channels.

Regarding the TCP protocol, it was adopted in our simulations the TCP Reno version [15], considering that it is one of

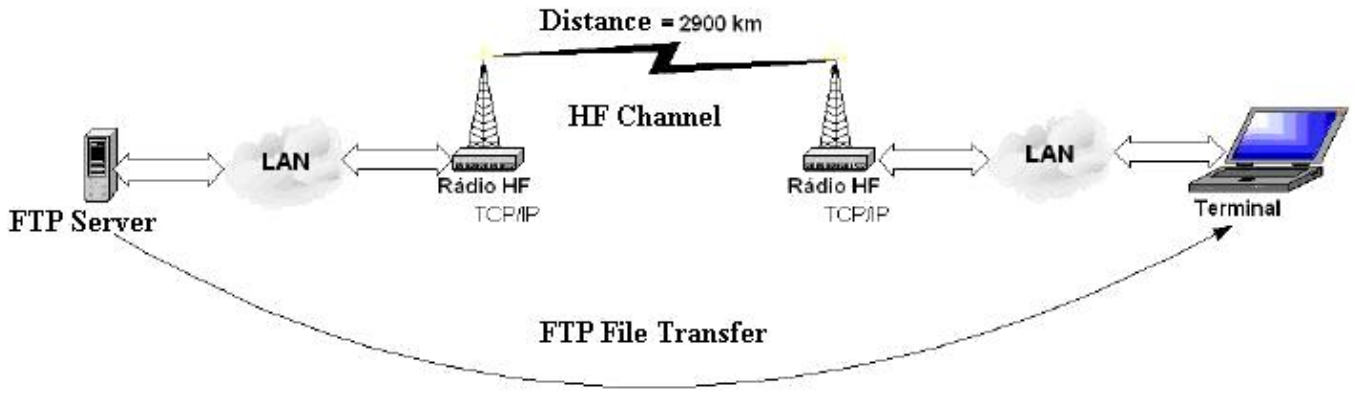

Fig. 1. Communications scnerario here investigated.

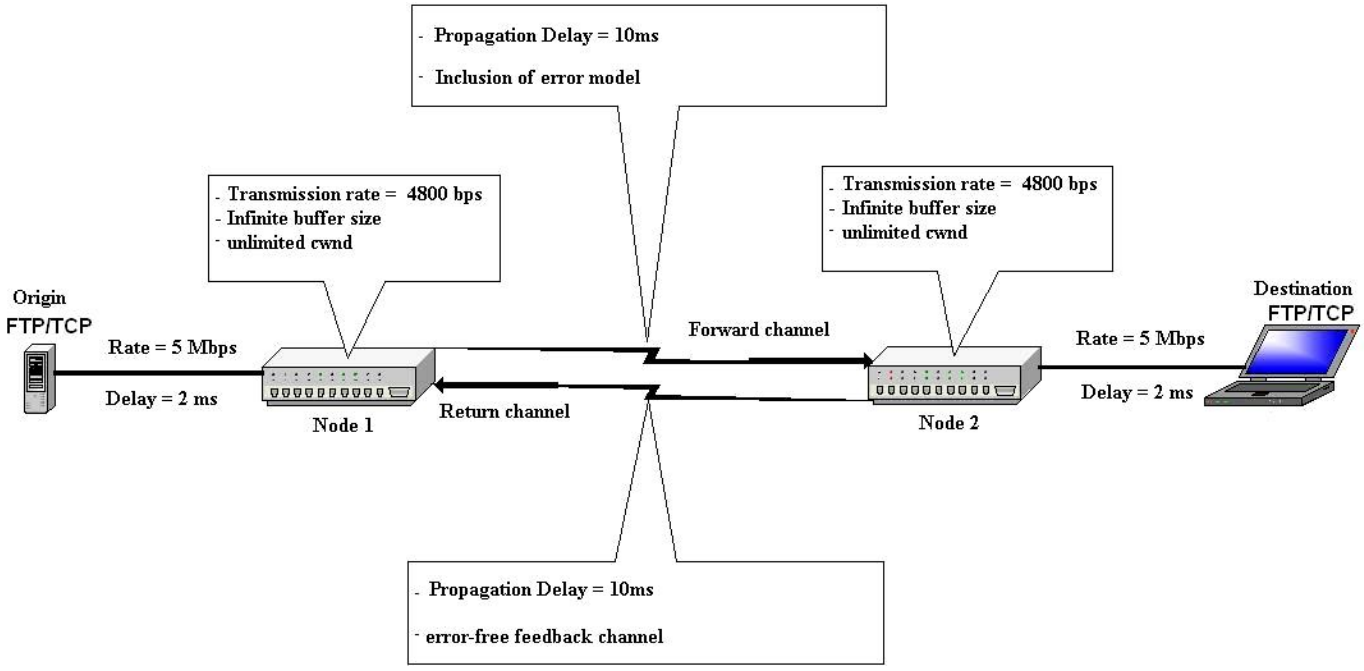

Fig. 2. Simulation model implemented in ns-2.

the most widely adopted TCP schemes. Besides, the reception window (*awnd*) of HF gateways are assumed to be of infinite size, so that the losses were caused only by channel errors. Thus, the sizes of the transmission and congestion windows are identical and the manipulation of *cwnd* directly influence the rate of segment transmission. In addition, the congestion window size *cwnd* is unlimited.

B. Bit Error Model for a HF channel

Some previous investigations have shown that the performance of data link layer and transport layer protocols in wireless channels depends to a great extent on the error model adopted for the physical layer. In [4] a compilation of recent work involving error modeling shows that the statistical characteristics of the channel errors have significant impact on the performance of communication protocols. In particular, it highlights the impact of error correlation on the performance of those protocols.

Therefore, a simple specification of the average error rate does not provide enough information for proper performance evaluation. As an example, it was shown in [7] that, for the same average rate of errors, the probability of a block of bits being successfully received can be doubled, depending on the auto-correlation function of the error process. This leads us to conclude that the investigation and use of accurate statistical models of the bit-error process are of fundamental importance to evaluate the performance of higher-level protocols in these communications scenarios.

In [4] it is shown that a Markov Model with 2 (two) states can fit the packet loss due to buffer overflow on systems with limited resources. Markov Models with a larger number of states may provide more accuracy in the modeling of those processes, at the price of increases in complexity of analysis and computational burden for parameter adjustment.

Fritchman [6] proposed to model error processes by Markov chains with several states partitioned into two groups ofe “error

free" states and "error" states. Fritchman models have been applied by many researchers to errors produced in several transmission systems over fading channels.

In order to better model the burst errors that typically occur in ionospheric HF channels a Hidden Markov Model (HMM) with three states was adopted in this work. In respect of the matrix of state-transition probabilities, we adopted in this model the one presented in [9], which is given by:

$$M = \begin{pmatrix} 0.99911 & 0 & 0.00089 \\ 0 & 0.73644 & 0.26356 \\ 0.36258 & 0.58510 & 0.05232 \end{pmatrix} \quad (6)$$

This matrix was estimated from data collected in an HF link between Chicago and San Diego by *ITT Research Institute* [9].

In the present work, the other parameters of the HMM model (conditional probabilities of error given a state) were set in order to fit some pre-established conditions of target bit error probability (P_{target}) and Doppler spread in the HF propagation channel, which are presented in the following section.

For simulation purposes, the fitted HMM models were implemented in C program to generate bit error traces that have been mapped into packet error traces of a given size. These later traces have been loaded in the ns-2 Simulator for TCP-based performance evaluation, as illustrated in Fig. 3.

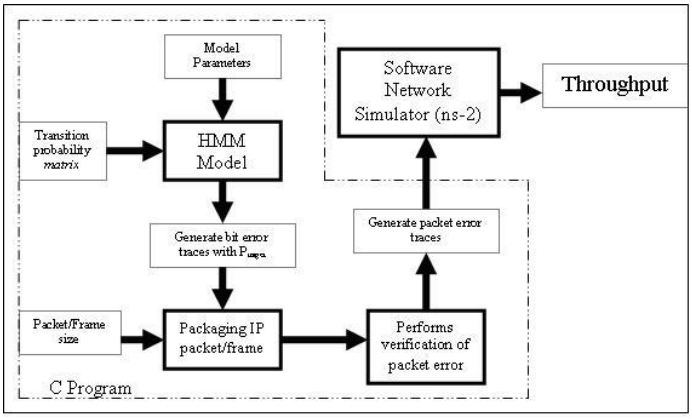

Fig. 3. Block diagram of the simulation environment.

LAYER	PARAMETERS	VALUE
Transport	FTP File Size	10000 bytes
	TCP version	Reno
	TCP/IP Packet Header	40 bytes
	Minimum Timeout	20ms
	Initial *cwnd*	1
Data-Link	Frame Size	50 bytes
	Frame Header Size	6 bytes
	Persistency δ	3 retransmissions
	NACK Timer	834ms (time necessary to send 10 Idle Frames)
Physical	Transmission Rate	4800 bps
	FEC Rate	1 (no coding) or $1/2$
	Target Probability (P_{target})	10^{-3} (no coding) or 10^{-5} (Rate $1/2$)

VII. Results

A. Comparison of strategies for improving TCP Throughput over HF Channels

In order to perform an initial investigation of strategies for improving the TCP over HF channels, we considered the set of parameters shown in Table VII-A.

The frame-header length of data-link protocol was set to 6 bytes, which corresponds to the AX.25 header length (20 bytes) without the address field (14 bytes). This value was adopted because the link is supposed to be point-to-point, so there is no need for including the address field in the protocol header.

Fig. 4 presents curves of throughput (B) as a function of the IP packet size. If the size of a transmitted FTP file in bits is denoted by F and T denotes the total time of delivery to the destination in seconds, the throughput is given by:

$$B = \frac{F}{T} \quad (bits\ per\ second) \quad (7)$$

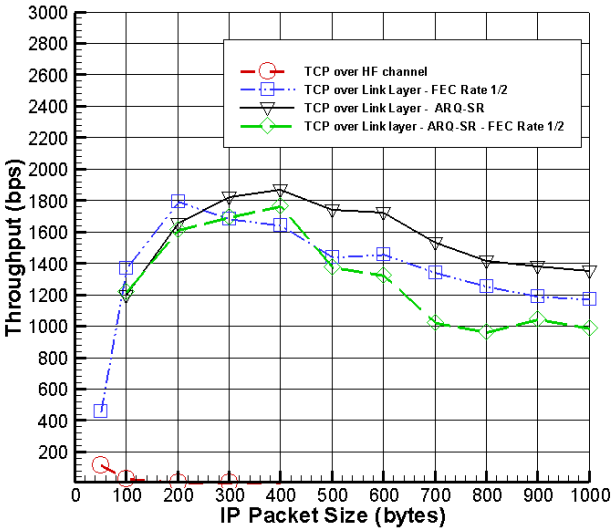

Fig. 4. TCP/IP throughput as a function of the IP packet size, for different strategies or error control at the link-layer.

As expected, Fig. 4 shows that when no strategy is used for error control at the data link layer (curve labelled "TCP over HF channel") a poor performance is obtained. On the other hand, outstanding performance improvements are observed when such strategies are applied (curves whose labels begin with "TCP over Link Layer"). So, these results clearly illustrate the importance of employing error control techniques in the data link layers of HF subnets, in order to improve the TCP throughput performance.

In fact, the ARQ-SR protocol and FEC techniques provide higher degrees of channel reliability, which allow TCP to increase *cwnd* and avoid the successive *timeout* increments, improving this way its throughput.

The results in Fig. 4 also show that the use of an ARQ-SR protocol at the data link layer produced the best results among the considered strategies, especially for IP packet size above

300 bytes. They also indicate that the size of the IP packet should be carefully chosen in order to improve the throughput.

B. Influence of the parameters of the ARQ-SR protocol on TCP performance

We have also evaluated the influence of the parameters of the ARQ-SR protocol on TCP performance. For this purpose, we set the IP packet size to 400 bytes, given that the best results were obtained from ARQ-SR strategy, as shown in Fig 4. Moreover, as this analysis did not consider the effect of FEC in the link layer, the P_{target} value was set at 10^{-3}.

The results concerning the effect of the frame size on TCP performance are shown in Fig. 5. It is observed that as the frame size increases, the performance degrades considerably. For example, when the frame size is increased from 25 to 100 bytes, the throughput reduction reaches 82.4%, despite the higher overhead in the first case compared to the second.

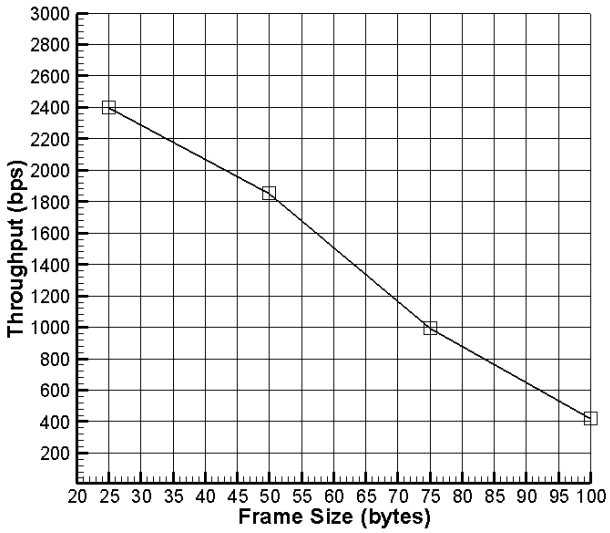

Fig. 5. TCP throughput as a function of the ARQ-SR frame size.

In face of these results, we can conclude that the performance of TCP protocol over HF channel in the presence of data link layer with ARQ-SR is extremely sensitive to the choice of the frame size.

Fig. 6 shows two curves of throughput versus IP packet size, obtained with ARQ-SR frame header sizes of 6 and 20 bytes. Considering that the frame size is 50 bytes, an increase in the header size from 6 to 20 bytes implies a reduction of about 31.8% in the data payload. For an IP packet size of 400 bytes, it may be observed in Fig. 6 a reduction in throughput performance of approximately 31%, showing the consistency of ours simulation results.

As presented in the previous sections, the value of the NACK *timeout* (Retransmit Time) that is here denoted by t_{nak}, has direct influence on the recovering speed of erroneous or lost frames at the reception side.

Fig 7 shows some results on the influence of the NACK *timeout* of the data-link protocol in the TCP throughput

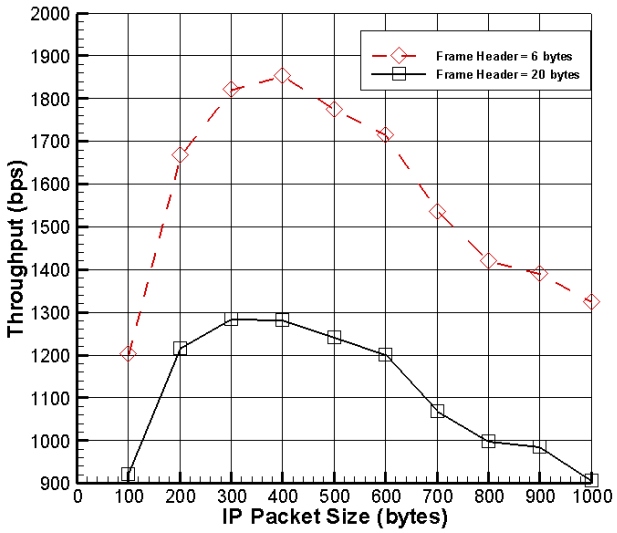

Fig. 6. TCP throughput versus IP packet size, for two values of the ARQ-SR frame header size.

performance. It should be noticed that in our implementation of the data-link protocol the Retransmit Timeout correspond to the number of successive transmissions of idle frames (control frames) before transmitting the NACK frame. Hence, for instance, a NACK timer of 10 corresponds to 834ms.

Fig. 7. TCP throughput as a function of the ARQ-SR NACK *timeouts*.

The results in Fig. 7 show that for $t_{nak} = 5$ the throughput obtained is maximum. The reduced throughput observed for values of $t_{nak} > 5$ is caused the increased delay in recovering frames with errors. On the other hand, the performance degradation for values of $t_{nak} < 5$ may be explained by excessively fast returns of NACK frames from the receiver, which cause excessive retransmissions of a same frame and thereby reduce

throughput.

Finally, the impact of the ARQ-SR persistency parameter δ on TCP performance was evaluated. Fig. 8 shows that when the value of δ is increased up to 3 there are significant improvements in TCP throughput. For higher values of δ, there is practically no change in the TCP throughput. This behavior suggests that for to the channel model and frame size adopted, virtually all the NACK occurrences of a same frame have been solved up to the 3-th transmission attempt.

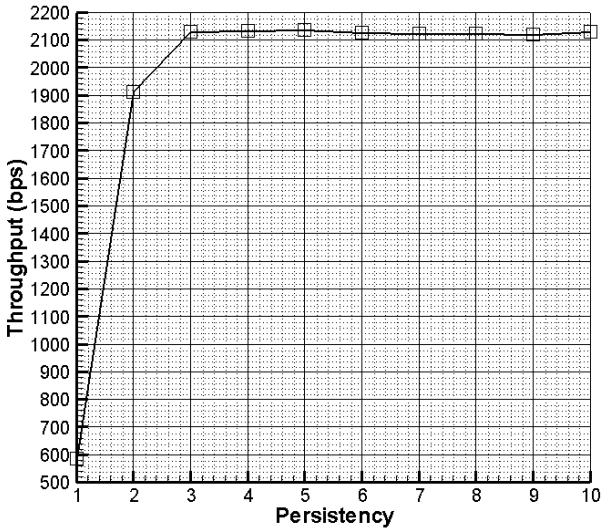

Fig. 8. TCP throughput as a function of the ARQ-SR persistency parameter.

C. The tuning of ARQ-SR parameters as a tool to improve TCP performance

After evaluating the isolated influence of these ARQ-SR parameters on the TCP throughput, we have also evaluated the effect of using the set of parameter values shown in Table VII-C. These values have been chosen on the basis of the previous results concerning the isolated impact of each parameter.

LAYER	PARAMETERS	VALUE
Transport	FTP File Size	10000 bytes
	TCP version	Reno
	TCP/IP Packet Header	40 bytes
	Minimum Timeout	20ms
	Initial *cwnd*	1
Data-Link	Frame Size	25 bytes
	Frame Size Header	6 bytes
	Persistency	3 retransmissions
	NACK Timer	417ms (time necessary to send 5 Idle Frames)
Physical	Transmission Rate	4800 bps
	FEC Rate	$1/2$
	Target Probability (P_{target})	10^{-3}

For the sake of comparison, Fig. 9 shows the results of TCP throughput versus IP packet size so obtained, as well as similar results that have been previous obtained with the parameter values of Table VII-A.

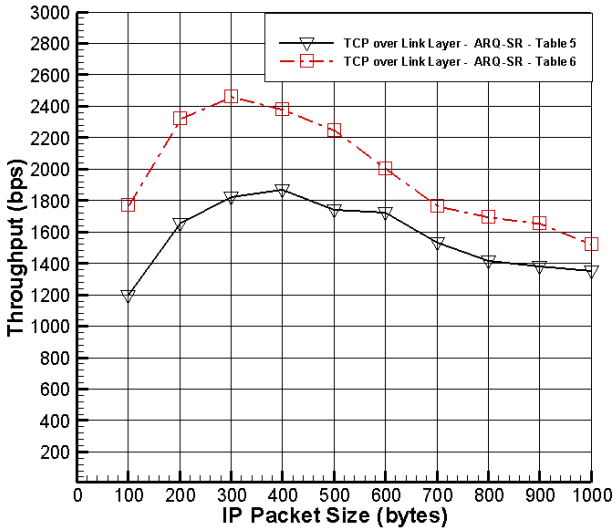

Fig. 9. TCP throughput versus IP packet size using a data link layer with ARQ-SR protocol configured in accordance with tables VII-A and VII-C.

This figure clearly illustrates the performance improvements produced by the use of a well chosen of ARQ-SR parameters. It should be noticed that for an IP packet size of 300 bytes, the throughput improvement is approximately 26%. A comparison of tables VII-A and VII-C shows that this performance improvement is due to the changes in the values of frame size and NACK *timeout*.

VIII. Conclusions

The impact of lower-layer countermeasures on the TCP throughput performance over HF channels was addressed in this work, where a simulation-based performance investigation of the use of a ARQ-SR protocol, a FEC technique and a hybrid ARQ-SR/FEC has been performed. The results showed the significant throughput improvement produced by these countermeasures. They also showed that the use of ARQ-SR protocol produced the largest throughput gains.

Our results have also shown that the IP packet size is an important parameter to be set in the TCP protocol for improving its throughput over HF subnets.

We also investigated the effect of ARQ-SR protocol parameters on the TCP performance in HF channels. In more specific terms, we evaluated the effect of frame size, frame header size and NACK *timeout* on the TCP throughput. The effect of ARQ-SR persistency was also investigated. After a careful choice of these parameters, a significant performance improvement was obtained.

Summing up, the results here presented clearly suggest that the parameters of transport and data-link layers and their interactions strongly affect the TCP performance over HF channels. Thus, the use of cross-layer techniques is a research topic of great interest in this context, which will be addressed in future works.

REFERENCES

[1] E.E. Johnson, "Third-generation Technologies for HF radio networking", *Military Communications Conference - MILCOM 98 Proceedings*, vol. 6, pp. 32-40, October 1999.

[2] H. Balakrishnan, V. N. Padmanabhan, S. Seshan and R. H. Katz, "A comparison of Mechanisms for Improving TCP Performance over Wireless Links", *IEEE/ACM Trans.Net.*, vol. 5, pp. 756-769, Dec. 1997.

[3] V. Tsaoussidis and I. Matta, "Open Issues on TCP for Mobile Computing", *Wireless Comm. and Mobile Comp.*, vol. 2, pp. 3-20, Feb. 2002.

[4] M. Zorzi and R. Rao, "Perspectives an the Impact of Error Statistics on Protocols for Wireless Networks", *Personal Communications IEEE*, vol. 6, pp. 32-40, October 1999.

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

[6] B. Fritchman , "A Binary Channel Characterization using Partitioned Markov Chains", *IEEE Trans. on Information Theory*, vol. 13, pp. 221-227, April 1967.

[7] P. Dietrich and R. R. Rao, "Second order bounds on block error probabilities in stationary, time varying channels", *Proc. 1996 CISS*, pp. 1996, March 1996.

[8] M.C. Chain and R. Ramjee, "TCP/IP Performance over 3G Wireless Links with Rate and Delay Variation", *Proc. ACM Mobicom'02*, Setember 2002.

[9] , S. Tsai, "Markov Characterization Of The HF Channel", *IEEE Trans. on Communication Technology*, vol. 17, pp. 24-32, February 1969.

[10] MIL-STD-188-110B, "Interoperability and Performance Standards for Data Modems" *Technical Report, Department of Defense U.S.*, April 2000.

[11] NATO Standardization Agreement: Profile for HF Radio Data Communications, STANAG 5066, version 1.2.

[12] D.J. Brown ,S.E. Trinder and A.F.R. Gillespie, "An analysis of HF for IP sub-networks" , *HF Radio Systems and Techniques*, pp. 281, Jul 2000.

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

[14] R. Abdelmounmen, M. Malli, and C. Barakat, "Analysis of TCP Latency over Wireless Links supporting FEC/ARQ-SR for Error Recovery", *IEEE Communications Society*, vol. 7, pp. 3994-3998, Jun 2004.

[15] K. Fall and S. Floyd, "Simulation-Based Comparisons of Tahoe, Reno and SACK TCP" , *Comp. Commun. Rev.*, vol. 26, no. 3, pp. 5-21, July 1996.

[16] W. R Stevens, *TCP/IP illustrated, Volume I The Protocols*. Addison Wesley, February 2000.

Márcio Barroso Toscano Dantas was born in São Paulo, in June 1975. Received the Diploma in Communications Engineeering and M.Sc. degree from Military Institute of Engineering (IME), Rio de Janeiro, Brazil, in 2000 and 2006, respectively.

He is currently working toward DSc. degree in Defense Engineeering, the Military Institute of Engineering (IME), since 2010. His research interests include the areas of communications and networking with emphasis on cross-layer analysis and design, and TCP performance over wireless networks.

Juraci Ferreira Galdino received the B.Sc. degree from the Federal University of Paraíba (UFPB), Campina Grande, Brazil, in 1991, in electrical engineering, the M.Sc. degree from Military Institute of Engineering (IME), Rio de Janeiro, Brazil, in 1997, and the Ph.D. degree in from Federal University of Campina Grande (UFCG), Campina Grande, Paraíba, Brazil, in 2002, both in signal processing. Its research interesting are adaptive modulation, adaptive filtering, cross-layer and wireless communication systems. He joined the Electrical Engineering Department of IME, Rio de Janeiro, in 2003, and the Defense Engineering Graduation Program of IME, in 2007, where he is an Associate Professor.

Ernesto Leite Pinto was born in Itaporanga, Paraíba, Brazil, in 1960. He received the Electrical Engineer degree from the Universidade Federal da Paraíba, Brazil, in 1983, the M.S. and Ph.D. degrees from the Pontifícia Universidade Católica do Rio de Janeiro, Brazil, in 1986 and 1998, respectively. He is an Associate Professor of the Instituto Militar de Engenharia (IME), Brazil, where he has been working since 1987. His research interests include channel modeling, performance evaluation of digital transmission systems, and advanced modulation and receiving techniques for multipath channels. Dr. Pinto is a member of the Sociedade Brasileira de Telecomunicações.