Comparison of FEC Types with Regard to the Efficiency of TCP Connections over AWGN Satellite Channels

Nedo Celandroni

Abstract— **Optimizing the end-to-end throughput of a TCP connection (goodput) over geostationary satellite links is a challenging research topic. This is because the high delay-bandwidth product, together with a non-negligible random loss of packets, is a condition that considerably differs from the environments TCP was originally designed for. As a result, TCP performance is significantly impaired by the channel bit error rate. The literature is full of suggestions for improving TCP goodput, most based on modifications of the protocol itself. We only investigated the application of different FEC (forward error correction) types for TCP goodput optimization, leaving the end-to-end protocol unaltered. Using a method midway between analysis and simulation to evaluate the goodput of TCP long-lived connections, we first studied the influence of packet loss rate, introduced by errors on the channel, on the TCP goodput. We showed that, in some cases, the packet loss rate does not need to be negligible with respect to that caused by congestion, as it is widely-held opinion. We then applied physical-level FEC techniques, such as convolutional encoding/Viterbi decoding, Reed Solomon, link-level erasure codes and their combinations, over a wide field of signal to noise conditions of the satellite channel. For each FEC type, we found the FEC rate that maximizes the TCP goodput, in each channel condition. We finally compared the results of all FECs used between them, and presented the case of multiple TCP connections sharing the same link as well.**

Index Terms— **Satellite link, TCP goodput, AWGN channel, FEC, BER, random packet loss.**

I. Introduction

THE transmission Control Protocol (TCP) is a connection-oriented, end-to-end, reliable transport protocol working between hosts in packet-switched networks of any possible topology. TCP is designed to operate over a wide spectrum of communication systems, ranging from wired to wireless and satellite networks. However, TCP performance may be severely impaired in environments where a high delay-bandwidth product is associated with a non-negligible packet loss due to data corruption. This occurs, for example, in satellite networks. Research in this area has long since highlighted this problem, and the literature is rich in both the performance evaluation area and in suggestions for improvement (see [1]–[7]) and in general the work carried out by the PILC IETF Working Group).

The performance problems of TCP over satellite links are becoming increasingly important as satellites become more widespread and their capacity increases. We foresee a growing importance of satellite connections in the future, due to some of their intrinsic advantages over wired links, such as: ease of scalability (new user installations), resilience to terrestrial damage, and multicasting efficiency.

Transmitting TCP traffic over satellite links rises a problem that seldom appears on terrestrial links. TCP behaviour is very sensitive to packet loss, which is interpreted as a congestion signal, and consequently as a reason to throttle the data rate. The task of the data link is to discard corrupted packets, which are not made available to the superjacent TCP/IP stack. This means that the TCP/IP stack cannot distinguish losses due to data corruption from losses due to other reasons that are interpreted as congestion signals. In order to increase the efficiency of TCP over satellite links, where improving the error rate is generally expensive (when possible), a number of techniques are commonly used. These include various types of spoofers, which may or may not preserve the TCP end-to-end semantics [8], with relative drawbacks [5]. Other techniques concern automatic repeat request (ARQ) [9]. These techniques, each operating at different levels of the protocol stack, exploit local knowledge of the satellite link characteristics in order to add a shorter delay control loop underneath the end-to-end control loop in the TCP connection. This gives a quicker reaction to packet loss and consequently improves the end-to-end performance at the expense of local buffering of packets to retransmit in case of loss and increased complexity. Methods have been proposed for a number of different techniques operating at different levels. For example, link-level forward error correction (FEC) [10] operates below TCP. Some methods, such as explicit loss notification (ELN) [11], which make the TCP stack aware of packet losses due to link errors, need some sort of cross-layer actions between TCP and the link level, and require TCP modifications at the end points. Other methods proposed involve changing the TCP stack at one or both end points, for example TCP Westwood [12] and its variants or TCP-Peach [13].

Reference [14] describes a technique that operates at the physical level, by trading the satellite channel bandwidth for packet loss rate. It does not interfere in any way with the normal behavior of the TCP stack, but simply entails that the wireless link parameters be appropriately tuned at the physical or link levels. In [14] it is argued that, given an

Manuscript received April 6, 2004; revised January 5, 2005 and May 31, 2005; accepted July 19, 2005. The associate editor coordinating the review of this paper and approving it for publication was W. Liao. This work was supported by the EC in the framework of the SatNEx NoE (Contract n. 507052) and by the Italian MIUR in the framework of the IS-MANET project.

N. Celandroni is with the CNR-ISTI Institute Pisa 56124, Italy (e-mail: nedo.celandroni@isti.cnr.it)

Digital Object Identifier 10.1109/TWC.2006.xxxxx.

1536-1276/06$20.00 © 2006 IEEE

available radio spectrum, antenna size and transmission power, the selection of a modulation scheme and a FEC type allows choosing both the bit error rate (BER) and information bit rate (IBR) of the link that maximize the throughput of a TCP connection, and thus the end-to-end transfer rate (also called goodput). This optimization can be made for different channel quality conditions whose variability is due, for instance, to the variable atmospheric attenuation of the signal. The optimal transmission parameters for each channel condition can be stored in look-up tables and dynamically applied in an adaptive fashion.

In the numerical example reported in [14], a 1/2 rate convolutional encoding/Viterbi decoding technique is used, with puncturing in order to obtain more coding rates. The modem has the ability to switch between BPSK (binary phase shift keying) and Q (quadrature)PSK. A similar approach is followed in [10], where the FEC is performed at link-level by using a block erasure code and packet losses both independent and correlated [15] are considered. In this paper we assume, as in [14], to be in the presence of additive white Gaussian noise (AWGN) so that we can consider independent packet losses. This assumption is reasonable when operating with geostationary satellite links and fixed user antennas. Here we extend the results obtained in [14], by introducing Reed Solomon, erasure codes and mixed techniques (concatenated codes). A comprehensive scenario of the various techniques, including the one used in [10], is produced for different channel quality conditions, and the relevant comparison is thus made possible. In other words, we wanted to discover which FEC type and rate, among some widely used ones, make the millions of TCP installations perform better on various conditions of satellite links. The results obtained can be helpful both in designing new network architectures and in improving already existing ones; for example, by adding an outer code, such as an erasure type one, which can be implemented even by software at link layer. The case in which multiple TCP connections share the same link was presented as well. In order to evaluate the goodput of TCP connections, we developed a fluid simulator tool, which is described in Section II. In Section III we give an overview of the FEC types used, and the TCP performance obtained by using them are shown in Section IV. Conclusions and the evolution of the present work are illustrated in Section V.

II. TCP GOODPUT EVALUATION

An analytical model for TCP is here presented, which is valid for high delay-bandwidth products and for one or more connections that share the same link. The purpose of the model is to develop a fast fluid simulator that we called TGEP (TCP goodput evaluation program). TGEP has been useful both to make the huge number of simulation runs, required to obtain the desired results, in a reasonable time, and to have one more tool, other than *ns2*, in order to validate results. We chose the Reno version of TCP. Other versions, such as TCP NewReno and TCP SACK, perform better in recovering multiple losses per RTT (round trip time); however, for the purpose of our study, that is the maximization of the goodput, the field of interest is relative to high TCP efficiency and in these conditions we found (by using *ns2* simulator) that all the three versions have practically the same behaviour. TCP Reno, new Reno and SACK together are the majority of all TCP implementations in the world.

The TCP Reno congestion control mechanism description can be found in [14], [16] and [17].

A. TCP Model

The following fluid model is valid for high delay-bandwidth product links, i.e. for connections with an average flight size of at least 10 segments. We suppose segment losses are due to both congestion and data corruption. We consider a Reno TCP implementation without the SACK [18] and with the Window Scale [19] options, so that the receiver advertised window (*rwnd*) never limits the flight size, and the transmission window is always equal to the congestion window *cwnd*. The analysis considers long-lived connections where the length of the *Slow Start* phase is negligible with respect to the length of *Congestion Avoidance* phases. Further assumptions are that the transmitting side always has data to transmit, and that all packet loss detections occur as the result of receiving three duplicate ACKs, never because of a retransmission timeout.

As in [16], in our system a buffer of capacity B is associated with the bottleneck link, whose transmission rate is μ segments/s. Denoting by T the minimum round-trip delay between a segment sending and the reception of the relative ACK, which consists of the double link latency τ plus the segment service time; we have $T = \tau + 1/\mu$. We also define the *normalized buffer size* $\beta = B/(\mu T)$ and denote by ω the congestion window size *cwnd* expressed in segments. We define a *flow control cycle* as the system evolution between the beginning of two consecutive *Congestion Avoidance* phases; during such a cycle, ω goes from a value ω_s to a value ω_e that is reached when a segment loss is detected; that is, when a set of four equally-numbered ACKs is received. If we neglect the temporary *cwnd* inflation during the *Fast Recovery* phase [17], we can observe that ω reaches the value $\mu T + B$, that we denote by ω_{max}, when the segment loss is due to congestion, while it reaches lower values when the loss is due to (random) data corruption. In the steady-state analysis of the TCP behavior, we consider only the *Congestion Avoidance* and *Fast Retransmit/Recovery* phases. This limitation is allowable when connections are long enough and timeout events are rare.

1) Single connection per link: In *Congestion Avoidance*, the window size ω is increased by the TCP sender by a value $1/\omega$ on each ACK reception. Denoting by b the number of segments acknowledged by each ACK ($b = 2$ if the delayed ACKs algorithm is used) we have [20] the following approximated relation:

$$\frac{d\omega(t)}{dt} = \frac{1}{\omega}\frac{da}{dt}$$

where da/dt is the arrival rate of the ACKs. If $\omega \leq \mu T$, $da/dt = \omega/(bT)$, otherwise $da/dt = \mu/b$. Thus, we have

$$\frac{d\omega(t)}{dt} = \begin{cases} 1/(bT), & \omega \leq \mu T \\ \mu/(b\omega), & \omega \geq \mu T \end{cases}$$

In the absence of data corruption losses, the bottleneck link buffer remains empty during the first phase (linear window-growing phase); then the buffer starts filling up to the value

B, when a segment is lost due to buffer overflow (congestion); after that, the window is halved, according to the AIMD (additive increase multiplicative decrease) congestion control.

2) Multiple connections per link: Let us make all calculations for the general case in which multiple TCP connections share the same link. We consider they have the same link latency τ and segment loss rate q. We also assume that all links between TCP senders and the common bottleneck buffer have a much higher capacity than that of the bottleneck link, and all connections have the same segment size. In this case, we denote by N the number of TCP connections and by ω the value of the aggregated $cwnd$, which is the sum of the $cwnd$s of all connections, while we add the index i to the variables denoting the values relevant to each individual connection. The single connection case is deducible by setting $N = 1$ and the indexed variables equal to the non-indexed ones.

Let us consider the $B \leq \mu T$ $(\beta \leq 1)$ case. For $\omega \leq \mu T$, the window evolution of the connection i is

$$\frac{d\omega^{(i)}(t)}{dt} = \frac{1}{\omega^{(i)}}\frac{da^{(i)}}{dt} = \frac{1}{\omega^{(i)}}\frac{\omega^{(i)}}{bT} = \frac{1}{bT} \tag{1}$$

Relation (1) produces

$$\omega^{(i)}(t) = \omega_s^{(i)} + t/(bT) \tag{2}$$

and adding up both sides of (2) for all connections, we get

$$\omega(t) = \omega_s + \frac{N}{bT}t. \tag{3}$$

As ω segments are sent in each interval T, denoting by $n(t)$ the total number of segments sent from the beginning of the cycle by all connections, and considering (3), we have

$$n(t) = \int_0^t \frac{\omega(z)}{T}dz = \frac{\omega_s}{T}t + \frac{N}{2bT^2}t^2 \tag{4}$$

which, inverted, gives

$$t(n) = bT\left(\sqrt{\omega_s^2 + 2nN/b} - \omega_s\right)/N. \tag{5}$$

Analogously, starting from relation (2), the number of segments sent by the sender i is

$$n^{(i)}(t) = \frac{1}{T}\int_0^t \omega^{(i)}(z)dz = \omega_s^{(i)}t/T + t^2/(2bT^2). \tag{6}$$

Substituting t, given by (5), in (6) we get the number of segments sent by the sender i when n segments are totally injected into the bottleneck link.

For $\omega \geq \mu T$ (second phase) the window evolution of the connection i is

$$\frac{d\omega^{(i)}(t)}{dt} = \frac{1}{\omega^{(i)}}\frac{da^{(i)}}{dt} = \frac{1}{\omega^{(i)}}\frac{\mu}{b}\frac{\omega^{(i)}}{\omega} = \frac{\mu}{b\omega}. \tag{7}$$

In fact, the total ACK arrival rate is μ/b and $\omega^{(i)}/\omega$ is the share that arrives at the sender i. Adding up both sides of 7) for all connections we get $d\omega/dt = N\mu/(b\omega)$, which yields:

$$\omega(t) = \sqrt{\omega_s^2 + 2N\mu t/b}, \text{ and } \omega(n) = \sqrt{\omega_s^2 + 2Nn/b}. \tag{8}$$

In (8), ω_s is the initial value of the window in the second phase, while t and n are the time and the number of segments injected into the bottleneck link from the beginning of the phase, respectively. When the second phase follows the first one (i.e. the cycle begins in the first phase), $\omega_s = \mu T$. In the second phase, the rate of segment sending is μ, so the number of segments sent is

$$n(t) = \mu t. \tag{9}$$

Denoting by t_a and t_b the temporal durations of the first and second phases when the cycle begins with $\omega_s \leq \mu T$, respectively, by setting in (3) $\omega(t_a) = \mu T$, we get $t_a = bT(\mu T - \omega_s)/N$, and from the first of (8), by setting $\omega(t_b) = \omega_{max} = \mu T(1+\beta)$, we get $t_b = b\mu T^2\beta(2+\beta)/(2N)$. Denoting by n_a and n_b the number of segments sent in the first and second phase, respectively, we get n_a from (4) as $n_a = n(t_a)$ and n_b from (9) as $n_b = n(t_b)$. Substituting $\omega(t)$ given by the first of (8) in (7), after integration we get:

$$\begin{aligned}
&\omega^{(i)}(t) = \omega_s^{(i)} + \left(\sqrt{\omega_s^2 + 2N\mu t/b} - \omega_s\right)\Big/N \\
&= \omega_s^{(i)} + \Delta\omega(t)/N, \text{where } \Delta\omega = \sqrt{\omega_s^2 + 2N\mu t/b} - \omega_s, \text{ and} \\
&\omega^{(i)}(n) = \omega_s^{(i)} + \left(\sqrt{\omega_s^2 + 2Nn/b} - \omega_s\right)\Big/N \\
&= \omega_s^{(i)} + \Delta\omega(n)/N
\end{aligned} \tag{10}$$

from which we see that the increment of the aggregated window, denoted by $\Delta\omega$, is evenly spread over all connections. Denoting by Q the number of segments in the buffer, the number of segments sent by each connection can be computed as

$$n^{(i)}(t) = \int_0^t \frac{\omega^{(i)}(z)}{T + Q(z)/\mu}dz = \mu\int_0^t \frac{\omega^{(i)}(z)}{\omega(z)}dz \tag{11}$$

Then, substituting the first of (8) and (10) in (11), after integration and considering (9), we get

$$n^{(i)}(n) = \left(n + \frac{\Delta\omega}{N\mu}(N\omega_s^{(i)} - \omega_s)\right)\Big/N \tag{12}$$

Let us consider the case in which the packet loss is due to congestion only. All cycles are equal and the starting value of the window in each cycle is $\omega_s^{(c)} = \omega_{\max}/2 = \mu T(1+\beta)/2$; thus, if $B \leq \mu T\,(\beta \leq 1)$, from (3) we get the duration of the first phase

$$t_a^{(c)} = \mu T^2(1-\beta)/(2N).$$

The average goodput can be computed as the sum of segments sent in a cycle divided by the temporal length of the cycle itself. Denoting by λ_c the goodput for congestion losses we have

$$\lambda_c = \frac{n_a^{(c)} + n_b}{t_a^{(c)} + t_b + N/\mu} = \frac{3b(1+\beta)^2\mu^3T^2}{4(2N^2 + b(1+\beta+\beta^2)\mu^2T^2)} \tag{13}$$

where $n_a^{(c)} = n(t_a^{(c)})$ is obtained from (4) with $\omega_s = \omega_s^{(c)}$.

The term N/μ in the denominator of (13) takes into account the time spent to resend the segments lost as a result of congestion. In fact, when congestion occurs, during an entire round-trip time, all connections experience a loss because, when the buffer is full, the ACK that triggers the transmission of two segments back-to-back causes an overflow, and it occurs for all connections.

Denoting by q_c the segment loss rate due to congestion, and imposing $n_a^{(c)} + n_b = (1 - q_c)/q_c$, we get

$$q_c = \frac{8N}{8N + 3b(1+\beta)^2\mu^2T^2}. \quad (14)$$

If $B \geq \mu T\,(\beta \geq 1)$, the linear window-growing phase does not occur, because $\omega_s \geq \mu T$; thus, relation (13) becomes

$$\lambda_c = \frac{3b(1+\beta)^2\mu^3T^2}{8N^2 + 3b(1+\beta)^2\mu^2T^2},$$

while the relation (14) is valid for any value of β.

We emphasize the importance of the buffer presence. In fact, from relation (13) we see that the normalized goodput λ_c/μ, for a single connection, no delayed ACKs and a value of $\tau = 0.5s$, which is typical for a geostationary satellite, is about 71% without the buffer ($\beta = 0$). Instead, a value of $\beta = 0.8$ is enough to reach 97% of the link rate (98% with $\beta = 1$), even for values of μ as low as 10 segments/s.

When the loss is due to data corruption, it occurs before ω reaches ω_{max}, and each cycle still begins with a value of ω_s, which is half the value attained in the previous cycle. In order to compute the average goodput in this case, we wrote a program (TGEP) whose source is available written in $\mathrm{Mathematica}^{\mathrm{TM}}$ language [21]. TGEP is a fluid simulator in that it draws the number of packets between two successive losses and computes analytically all system variables as they were continuous, without simulating packet by packet.

We assume the packet loss follow a binomial distribution with average loss rate q and denote by n_{max} the max. number of segments in each cycle; $n_{\max} = n_a + n_b$ if $\omega_s < \mu T$,

$$n_{\max} = \frac{b\left(\mu^2T^2(1+\beta)^2 - \omega_s^2\right)}{2N}$$

otherwise. Starting with the value ω_s at the beginning of each cycle, we draw the number n of successful segments between two successive losses and we check in which case it falls. We can have $n < n_{max}$, or $n \geq n_{max}$. In the first case, the cycle is prematurely ended by a random loss; otherwise, the cycle is ended by a congestion loss. In the first case, in order to discover which connection experienced the loss, we consider that the loss probability of each connection is equal to $\omega^{(i)}/\omega$. So we draw the connection i that experienced the loss and halve the value of its window only. If n falls in the second case, the cycle is ended by a congestion loss; thus, during the next round-trip time period, all connections experience a loss. In this case we halve the windows of all connections.

Each cycle begins with the linear or the second phase according to $\omega_s \leq \mu T$ or $\omega_s > \mu T$, respectively. We compute the time duration of each cycle by using relation (5) for the linear window-growing phase and relation (9) for the second phase, respectively. The cycle duration is incremented by the time required to resend one or N lost segments in the event of random or congestion loss, respectively. All cycle durations and the total number of segments sent in relevant cycles are accumulated in the variables t_{tot} and n_{tot}, respectively; then the average aggregated goodput λ is computed as $\lambda = n_{tot}/t_{tot}$. The number of segments sent by each connection is accumulated in the variables $n_{tot}^{(i)}$, by using relations (6) and (12), and the average individual goodputs are then computed as $\lambda^{(i)} = n_{tot}^{(i)}/t_{tot}$. In order to determine the confidence interval for the mean of the random variable λ, a number of independent tests are repeated until the mean falls within the desired interval, which is computed using parameters given by the Student-t distribution.

TABLE I
GOODPUT OF A SINGLE CONNECTION – $ns2$, RELATION (15) AND TGEP COMPARISON

TCP normalized goodput (β=0.8, τ=0.5s, μ=100 seg./s, delayed ACKs, no SACK). $ns2$ and TGEP output conf. intervals = $\pm$1% at 99% level

q	$ns2$ Reno	$ns2$ NewReno	Relat. (15) T_0=1s	Relat. (15) T_0=2s	TGEP
10^{-1}	0.034	0.036	0.030	0.022	0.058
10^{-2}	0.158	0.165	0.161	0.154	0.162
10^{-3}	0.546	0.554	0.534	0.532	0.547
10^{-4}	0.955	0.955	1.697	1.697	0.957
10^{-5}	0.988	0.989	-	-	0.991

B. TGEP validation

TGEP allows the estimation of the goodput within a chosen confidence interval at a chosen confidence level. The limit of TGEP is that it does not consider TCP timeouts and it does not model multiple losses within an RTT, but in cases in which $q \ll 1$, which are significant cases when the delay-bandwidth product is high, TGEP performs satisfactorily. Unless otherwise specified, all results of TGEP in the rest of the paper have been obtained with a confidence interval of $\pm 1\%$ at 99% level and by using the parameter $b = 2$ (delayed ACKs). In order to obviate the non-consideration of TCP timeouts, we can assume the expression (15), in which the relation that gives the sending rate for unlimited bandwidth links, taken from [22], is divided by μ for normalization and is multiplied by $(1 - q)$ to better approximate the goodput.

$$T_g = \frac{1-q}{\mu\left[\tau\sqrt{\frac{2bq}{3}} + T_o\min\left(1, 3\sqrt{\frac{3bq}{8}}\right)q(1+32q^2)\right]} \quad (15)$$

where T_0 is the value of the timeout. In Table I we report the comparison of TGEP, relation (15), and $ns2$ simulation outputs, for a single connection; in relation (15) the value of T_0 has been set to both 1s and 2s. In Table II we report the comparison of TGEP, relation (15) (for both T_0=1s and 2s) and $ns2$ simulation for five connections. We can observe a good agreement between $ns2$ and relation (15) for low values, and between $ns2$ and TGEP for high values of the goodput; this agrees with our expectations, because relation (15) considers TCP timeouts and unlimited bandwidth, while TGEP considers the bottleneck rate of the link and does not take timeouts into account. A suitable goodput threshold to use when selecting one or the other procedure can be assumed equal to 50% of the bottleneck rate, even if TGEP agrees with relation (15), and with $ns2$, for much lower values of the goodput as well.

C. TCP goodput with congestion and random losses

By using TGEP, we have computed the average normalized goodput λ/μ as a function of the ratio q/q_c, i.e. the ratio

TABLE II
GOODPUT OF 5 CONNECTIONS – $ns2$ AND TGEP COMPARISON

TCP Reno (no delayed ACKs) goodput of 5 connections sharing a link with a bottleneck rate μ=455 segments/s, τ=0.5s, β=0.8. Confidence intervals are at 99% level.

$ns2$ simulations							
q	Connect. #1	Connect. #2	Connect. #3	Connect. #4	Connect. #5	Total	Relation (15) T_0=2s
10^{-4}	0.198± 3.1%	0.199±2.5%	0.199±0.5%	0.196±2.2%	0.202±2.3%	0.994±0.03%	2.677
10^{-3}	0.165±2.4%	0.169±2.6%	0.164±4.3%	0.169±5.3%	0.169±1.3%	0.836±0.67%	0.839
10^{-2}	0.047±0.64%	0.047±2.3%	0.047±1.5%	0.047±3.1%	0.047±1.6%	0.235±0.4%	0.243
10^{-1}	0.009±3%	0.009±0.8%	0.009±0.7%	0.009±0.6%	0.009±2%	0.045±4%	0.035
TGEP outputs							
q	Connect. #1	Connect. #2	Connect. #3	Connect. #4	Connect. #5	Total	Relation (15) T_0=1s
10^{-4}	0.199±3.5%	0.198±0.6%	0.199±5.8%	0.199±3.7%	0.199±1.89%	0.994±0.1%	2.678
10^{-3}	0.169±5.1%	0.168±7.7%	0.166±8.2%	0.171±7.5%	0.171±7.2%	0.845±1%	0.843
10^{-2}	0.050±5.1%	0.051±14%	0.052±8.4%	0.049±6.7%	0.049±5.8%	0.250±1%	0.254
10^{-1}	0.015±1.3%	0.016±1.7%	0.016±1.6%	0.015±1.5%	0.016±1.9%	0.078±1%	0.048

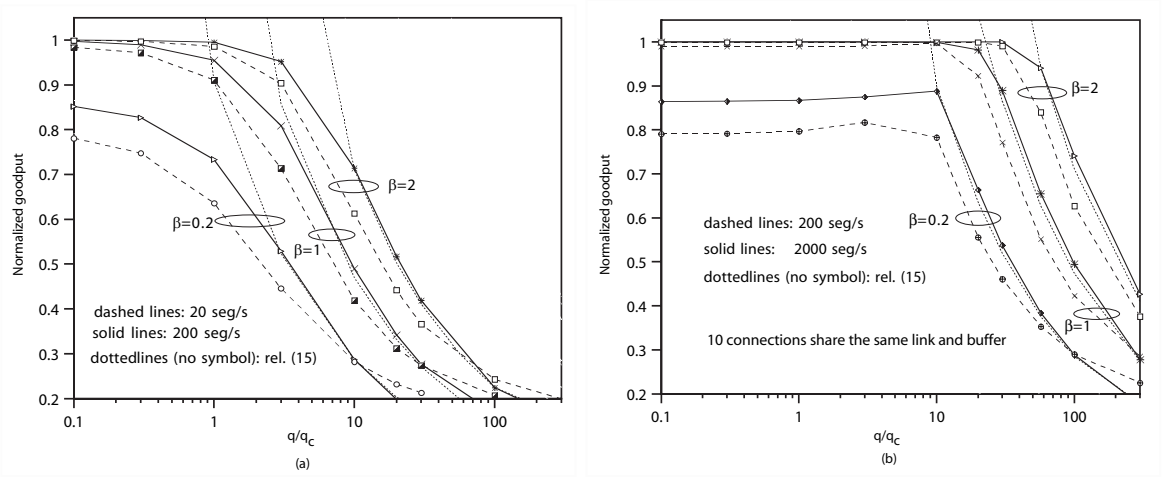

Fig. 1. TCP normalized goodput versus q/q_c ratio and $\tau = 0.5s$, for a single connection (a) and for 10 connections sharing the same link and buffer (b). Different values of the link rate μ and the normalized buffer size β are considered. Relation (15) outputs are also included for comparison.

between the loss rate due to segment corruption and the loss rate due to congestion only, given by (14).

We have reported the results in Fig. 1, for a single connection and ten connections that share the same link, τ=0.5, b=1 and for different values of β and μ; the values given by relation (15) are shown as well. The scenario depicted in Fig. 1 provides an insight into the question of a suitable limit for the packet loss rate due to data corruption in TCP connections with geostationary satellite links. The results show that the widely-held view in discussions about TCP performance on satellite links (see e.g. [2] and [7]), which entails making the packet loss rate due to data corruption always negligible with respect to loss due to congestion, should be partially reviewed. In fact, while in the single-connection case the goodput is always a decreasing function with the packet loss rate q, this is not true for multiple connections. In this case, the goodputs exhibit a maximum for values of q that are higher than the one relative to congestion only ($q/q_c > 1$). This is explained by the fact that in case of congestion, during a complete round trip, all connections experience a loss that halves all congestion windows (synchronization effect), with a consequent reduction in the aggregated goodput if the window values preceding the halving were not sufficiently high, i.e. for small buffer sizes. A moderate random loss rate, in this case, improves the situation because losses occur for a single connection at a time, before congestion can take place; thus, the aggregated congestion window remains at high values on average. It is worth noting that a similar benefit is obtained by using RED (random early detection) [23] instead of drop-tail as drop policy to discard packets in the bottleneck buffer. However, for a single connection, a noticeable degradation of the goodput is caused by a random loss rate that is higher than the congestion loss rate if the buffer is sufficiently large ($\beta \geq 1$ in the figure, $\beta \geq 0.8$ in practice). We point out that lowering the random loss, which means lowering the bit error rate (BER) over the link, is always expensive in terms of energy and/or information rate, that is the bottleneck rate itself. However, it is easy to infer that the calibration of the BER, i.e. the choice of the FEC and the modulation scheme to maximize the TCP goodput, admits an optimum value of the BER for each channel quality condition [14]. Figure 1 also highlights the approximation error of relation (15), which is often used, limited to 1, to evaluate the TCP throughput. The error relative to this approximation is lower for multiple

connections and decreases with the buffer size.

III. PERFORMANCE OF VARIOUS FEC TYPES

In this section we show the performance of several of the most widely-used FEC types in terms of segment error rate (SER), as a function of the signal-to-noise ratio available at an earth station receiver. We assume operating over an AWGN channel and a negligible inter-symbol interference, so we can assume that erred bits on the channel are independent. This hypothesis is widely accepted for users of geostationary satellites, equipped with fixed and directive antennas, while the user or satellite (in low earth orbits) mobility generally produces bursts of errors due to multi-path fading and/or shadowing.

A. Convolutional Coding

We used the characteristics of a convolutional coder/Viterbi decoder over BPSK/QPSK modulated symbols relative to the standard NASA 1/2 rate with constraint length 7 [24] and derived punctured codes, while the average error burst length (ebl) was obtained by using simulation. We refer to reference [14] for the complete set of data (BER and ebl) that is plotted versus E_c/N_0, that is, the ratio of channel bit energy to the one-sided noise spectral density.

B. Reed Solomon Coding

The Reed Solomon (RS) codes [25]– [28] are linear block codes with an alphabet size of $A = 2^m$, m being the number of bits in a symbol. An RS code is specified as RS(n, k), which means that k m-bit symbols of information are added to $r = n - k$ parity symbols to form an n-symbol codeword. The block (codeword) maximum length is $n = A-1$ symbols [26], but it can be extended to A and $A + 1$ [25]. If an RS code has r redundant symbols, the code is able to correct any pattern of t symbol errors and s symbol erasures for which $2t + s < d = r + 1$. An erasure occurs when the position of an erred (or missing) symbol is known, and d is the minimum Hamming distance between codewords. Thus, this code is able to correct up to $t_0 = r/2$ errors when there are no erasures and up to $s_0 = r$ erasures when there are no errors. The rate of an RS code is k/n; anyway, codes can be shortened by assuming sending a number z of zeroes (that are not obviously transmitted) and reinserting them at the decoder. This allows reducing the coding rate at will, which becomes $(k-z)/(n-z)$. Large values of r strongly decrease the block error rate, but require high computational power. Reed Solomon codes are particularly advantageous when used in the presence of error bursts as occurring on the output of a Viterbi decoder [14]. This is because an error burst of length ebl can corrupt s_i (the integer part of ebl/m) or $s_i + 1$ symbols at maximum. This makes the concatenation of RS (outer codes) with convolutional codes (inner codes) a very powerful FEC technique. The performance of such concatenated codes is enhanced by inserting (between the two coders) a symbol interleaver/de-interleaver, which reduces the correlation of the erred symbols. In this subsection we refer to unknown positions of erred symbols (RS codes), while we will analyze in the next Subsection the case of known positions of erred symbols (erasure codes). Denoting by p_s the symbol error probability, the block error probability p_b, in case of independent erred symbols (infinite interleaver depth), can be easily computed as

$$p_b = 1 - \sum_{j=0}^{t_0} \binom{n}{j} p_s^j (1-p_s)^{n-j}. \tag{16}$$

When erred bits can be considered as independent, e.g. in the case of an AWGN channel, the symbol error probability can be computed as $p_s = 1 - (1-p)^m$, p being the bit error probability (BER). In case of error bursts, such as at the output of a Viterbi decoder, simulation is required to estimate p_b. In the past, RS coding was always implemented in hardware; however, the computing power of today's machines allows software implementations as well [26].

One of the most popular codes is RS(255, 223) with m=8 and t_0=16. Choosing a segment length of 1115 bytes, which contains five 223-byte blocks, we evaluated the segment error rate (Fig. 2), for different values of E_c/N_0, and E_b/N_0 (information bit energy to the one-sided noise spectral density ratio), by using RS alone, convolutional encoder/Viterbi decoder with different coding rates, and concatenated RS/convolutional codes. The last ones are used both with and without an interleaver which, if present, is assumed to be ideal. An ideal interleaver allows us to consider independent erred symbols and relation (16) can be used.

An approximate evaluation of p_s, which is good for low values of p, is given by (see Appendix):

$$p_s = p(\overline{ebl} + m - 1)/\overline{ebl} \tag{17}$$

where $\overline{ebl}$ is the average error burst length. The SER is then given by $\mathrm{SER} = 1 - (1 - p_b)^{n_b}$, n_b being the number of blocks in a segment. For the simulation of the case without an interleaver we assumed (see [25] and reference therein) that the error burst length has a geometric distribution $\Pr\{ebl = l\} = p_l(1-p_l)^{l-1}; l = 1, 2, ..., p_l = 1/\overline{ebl}$, and the distance between error bursts has the empirical distribution $\Pr\{D = v\} = p_d(1-p_d)^{v-K-1}, v = K+1, K+2, ...$, where $p_d = 1/(\overline{D} - K + 2)$, $\overline{D}$ is the average distance between error bursts, which can be assumed as $\overline{D} = p/\overline{ebl}$, and K is the convolutional code constraint length. To compute the SER when the segment is convolutionally encoded, for small values of p, we can use the relation [14]: $\mathrm{SER} = 1 - (1 - p/\overline{ebl})^l$, where l is the segment bit length.

The evaluation of the code that performs better in energetic terms, given the SER value we want to achieve, is directly made by looking at the diagram in Fig. 2, which gives the SER versus E_b/N_0. Relations that link bit rates and signal-to-noise ratios are

$$R^{(i)} = C^{(n)} - E^{(b)};\ R^{(c)} = C^{(n)} - E^{(c)} \tag{18}$$

where we denoted by $R^{(i)}$ and $R^{(c)}$ the information and channel bit rates (expressed in dB), respectively, by $E^{(b)}$ and $E^{(c)}$ the values of E_b/N_0 and E_c/N_0, respectively, and by $C^{(n)}$ the value of C/N_0 (carrier power over one sided noise spectral density). From the first of (18) we see that, for a certain SER value, a code that needs a lower E_b/N_0 allows

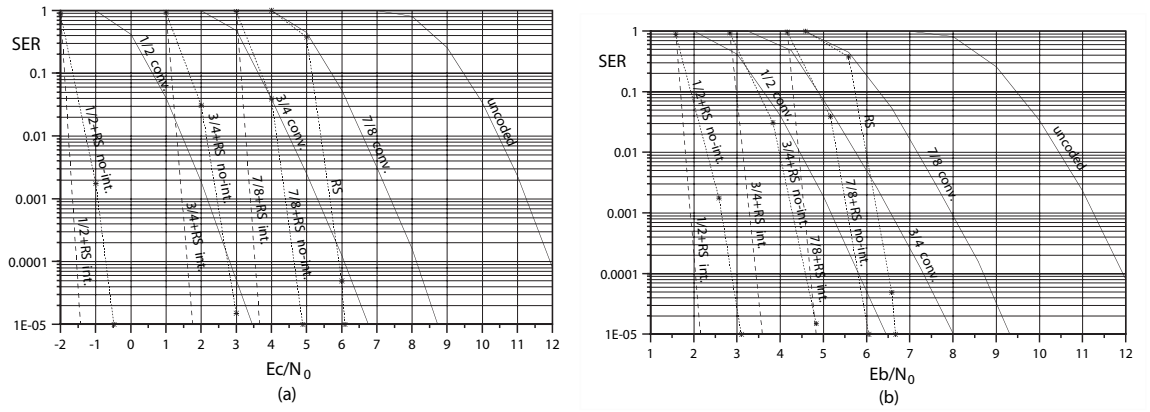

Fig. 2. SER versus E_c/N_0 (a), and versus E_b/N_0 (b), for different convolutional coding rates and for concatenated RS/convolutional codes, with and without symbol interleaver.

a higher information bit rate with the same C/N_0, if we assume having an ideal modem that is able to vary the bit rate continuously. As regards the spectrum efficiency, from the second equation of (18), a code that needs a higher E_c/N_0 allows a lower channel bit rate with the same C/N_0 and it occupies a narrower bandwidth. However, the comparison makes sense if the same information bit rate is obtained. Thus, introducing subscripts 1 and 2 for denoting two different codes, we have: $R_1^{(c)} = C_1^{(n)} - E_1^{(c)}$ and $R_2^{(c)} = C_2^{(n)} - E_2^{(c)}$; then: $R_1^{(c)} < R_2^{(c)}$ if $C_1^{(n)} - E_1^{(c)} < C_2^{(n)} - E_2^{(c)}$. To get the same $R^{(i)}$, $C_1^{(n)} - E_1^{(b)}$ must be equal to $C_2^{(n)} - E_2^{(b)}$; then $code_1$ requires a lower channel bit rate (lower spectrum occupancy), with respect to $code_2$ if $E_1^{(b)} - E_2^{(b)} < E_1^{(c)} - E_2^{(c)}$. As an example, the RS code has a spectrum occupancy that is lower than the 3/4 convolutional one, for a SER<0.001.

C. Erasure codes

The term erasure code refers to a block FEC that is able to recover from a corruption or loss of some blocks of information used for sending a data unit. An original piece of information (e.g. a TCP segment) is divided into k blocks and added to $r = n - k$ redundant blocks; the resulting n blocks are then individually sent in n frames. An erasure code that is easy to understand, flexible and efficient to implement, even on a common PC, is presented in [29], together with its implementation performance. In [10], an analysis of TCP throughput has been performed, by using FEC erasures over a lossy channel that is assumed to follow the Gilbert model [15]. Here, an AWGN channel is assumed with or without convolutional coding.

In the case of independent block losses, the SER can be computed by using relation (16) with p_s equal to the block loss rate (blr) and $t_0 = r$. In order to do this, we must know the position of the lost (or corrupted) blocks; thus, a simple protocol is needed in this case. For each block we assumed a cyclic redundancy check (CRC) field of 16 bits, a field to indicate the $block\_sequence\_number$ inside the packet, plus the $last\_block$ information and some spare bits for a 32 bit overhead. The performance of the erasure code, which we evaluated for independent bit errors, refers to 23, 46 and 92-byte block sizes and 48, 24 and 12 blocks per packet, respectively. Such metrics allows sending 1104-byte segments, which is very close to the size used in RS coding, thus allowing the performance comparison. In Fig. 3, E_b/N_0 and E_c/N_0 versus redundancy are reported for the three block sizes chosen and for a SER value of 10^{-4}. The same picture also shows both the channel and information bit rates, which would be possible to achieve with the same C/N_0, normalized with respect to the bit rate of the uncoded channel. For example, a segment sent with 48 23-byte blocks, with 16 redundant blocks, by using the same power as when sending an uncoded 1104-byte segment, can be sent at a channel bit rate of 3 times, and getting an information bit rate of 1.9 times, with respect to the uncoded segment, in order to achieve the same SER of 10^{-4}. We can observe that the minimum E_b/N_0

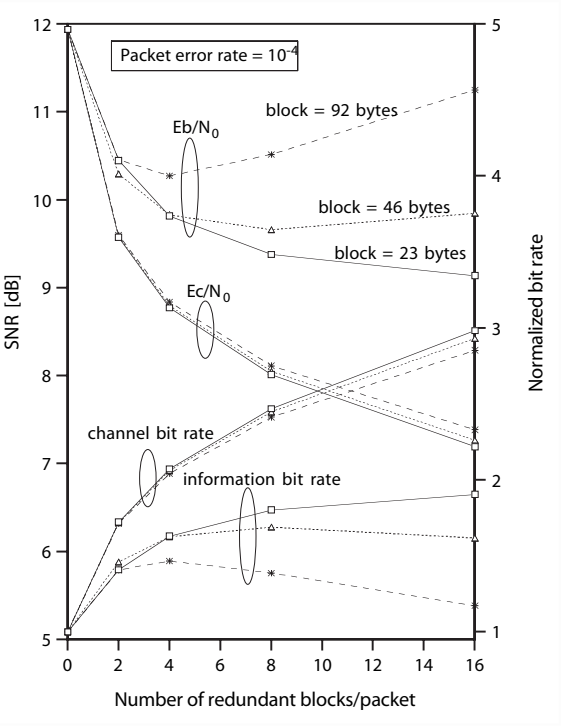

Fig. 3. Erasure codes: E_b/N_0, E_c/N_0, and achievable bit rates versus redundancy, for SER=10^{-4} and different block sizes.

is achieved in all cases with a redundancy of 1/3 and the 23-byte block case exhibits the best performance. The same test for different SER values gives similar results.

IV. TCP GOODPUT WITH DIFFERENT FECS AND MODULATION SCHEMES

Let us now consider our original problem, which is to find, for each channel condition, the code rate and modulation scheme that give the maximum TCP goodput for the various FEC types. It is worth noticing that the complexity of the procedure to evaluate the TCP goodput point by point does not significantly affect the applicability of the method. In fact, any analytical, as well as simulation or measurement-based methods, could be used to build look-up tables that will be employed during normal operation. In the case of dynamic bandwidth allocation, the optimization should be made both for all possible values of C/N_0 and for each possible per-user allocation of the capacity.

A. Analytical Considerations

Let us assume that a portion of a satellite transponder capacity is assigned to an earth station for a class of one or more TCP connections. This portion consists of a carrier that can be modulated, in phase and/or amplitude, at a rate of B symbols/s. We consider the widely diffused M-ary PSK or QAM (quadrature amplitude modulation) types. In these modulation schemes, M is the number of points, in the phase-amplitude space, relative to the constellation of the modulated symbols. Typical values of M are 2, 4, 8, 16, 32, and 64; BPSK and QPSK schemes correspond to M values of 2 and 4, respectively. We denote by c_r the resulting rate of codes applied to the base band bit stream and by ρ the actual portion of the carrier that has been assigned. ρ=1 if the transponder capacity is shared in FDMA (frequency division multiple access) mode (also called SCPC–single carrier per channel) and the whole carrier is thus assigned. Whereas, $\rho \leq 1$ in the case that the carrier is shared in TDMA (time division multiple access) mode and a time portion ρ has been assigned. The information bit rate, i.e. our bottleneck rate μ, is approximately given by

$$\mu = f_c \bar{B} \tag{19}$$

where we denoted by $f_c = \eta c_r$ the capacity factor, by $\eta = \log_2 M$ the spectrum modulation efficiency, and by $\bar{B} = \rho B$ the bandwidth actually assigned. Given a value of C/N_0, which ensues from a certain channel condition, for a modulation scheme and coding rate the operating E_b/N_0 is given by

$$E_b/N_0 = C/N_0 - 10Log_{10}\mu$$

according to the first of relations (18). Diagrams similar to the one shown in Fig. 2*b* are then used to estimate the SER, i.e. our q. Let us assume that each value of f_c is associated to the pair of parameters $\eta \in [\eta_{\min}, \eta_{\max}]$ and $c_r \in [c_{r_{\min}}, c_{r_{\max}}]$ that produce the minimum value of q, which thus monotonically increases with f_c. In order to analytically evaluate the maximum goodput, we would need an expression that gives the goodput as a function of f_c. Even if such an expression is not obtainable, we could resort to an interpolating function in the domain considered. Let $\bar{q}(f_c)$ be a continuous interpolating function that gives the packet loss rate for a certain channel quality (C/N_0), and let $T_g(\bar{\mu}, \bar{q})$ be the normalized goodput for fixed values of τ, buffer size and segment size, where $\bar{\mu}$ is given by relation (19) for a certain bandwidth allocation. The actual goodput is thus $G = \bar{\mu} T_g(\bar{\mu}, \bar{q})$ and its maximum value will be for a capacity factor (thus for a pair of η and c_r) such that $\partial G / \partial f_c = 0$, where the partial derivative can be assumed as the derivative of one variable function, all other variables being fixed. As an example, let us assume, for the normalized goodput, the approximate expression (15) limited to the unity. For low values of q, and for a given τ, the actual goodput is thus inversely proportional to $\sqrt{q}$, and limited superiorly by the bottleneck rate μ. Therefore, we have $G = \min\left\{k/\sqrt{q}, \mu\right\}$, where k is a constant. As q monotonically increases with f_c and then with μ, for a given channel condition and allocated bandwidth, the maximum goodput is obtained for the value of f_c such that $\bar{\mu} = f_c \bar{B} = k/\sqrt{\bar{q}(f_c)}$.

However, bit and coding rates are not variable with continuity, in practice; therefore, the values found analytically could not be used in most cases. Therefore, we prefer to evaluate the goodput by using simulation, for all FEC rate-modulation scheme pairs, then choosing the pair that produces the maximum value. This is done in next Subsections.

B. Single TCP Connection per Link - Simulation

We now estimate the goodput of a single TCP connection, for different channel quality conditions. We assume having a modem, working at 2 Msymbols/s (Ms/s), which is able to switch between BPSK and QPSK modulation schemes. Thus, the channel bit rates are 2 and 4Mb/s, respectively. This choice allows to represent a commonly used piece of equipment and to operate with the same spectrum occupancy in both cases. We also assume that the channel bandwidth is shared in TDMA mode among the users and that the considered TCP connection received an assignment of one-half the carrier capacity (ρ=1/2). The C/N_0 ratio at the demodulator ranges between 66 and 78 dB. C/N_0 depends on the signal attenuation, e.g. due to rain fading. The bit rate of uncoded data is thus 2 or 1Mb/s using QPSK or BPSK, respectively. The selection of the transmission parameters is made at the earth station on the basis of the link quality. The modulation scheme selection and convolutional coding/decoding is performed at the physical level; RS coding/decoding is preferably performed at the physical level (possibly by software at link level), while erasure coding/decoding is performed at link level [29]. If the satellite channel capacity is shared in FDMA mode among the users, a modulation rate of 1 Ms/s gives the same results as in TDMA mode. When a whole 2 Ms/s carrier is assigned to the TCP connection, the results are still valid when decrementing by 3 dB the range of C/N_0 ratio (63-75 dB).

Figure 4 ($a - f$) reports the results relative to the various coding types employed. All points lying on the envelope represent the maximum goodput obtainable with the available hardware for a given channel quality condition. In [14] it is shown that the adoption of a different criterion, such as the commonly-used one which keeps the BER below a given

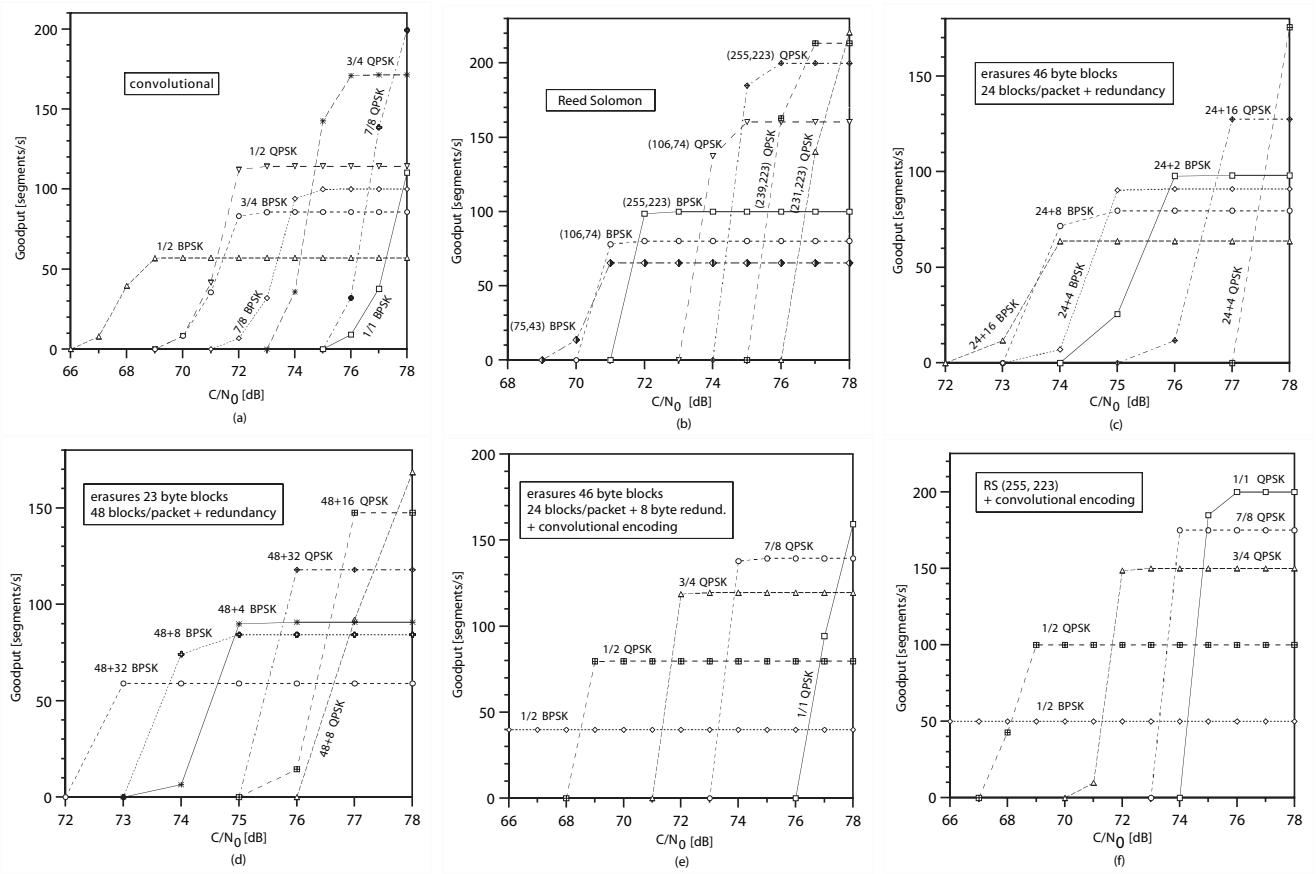

Fig. 4. TCP goodput versus C/N_0 for different coding types/rates and modulation schemes.

threshold, produces results that may considerably differ from optimality, even if the threshold is made variable with the link capacity. Our point-by-point optimization is more complex to estimate; however, the computation is made only once and it does not add any overhead at the running time. Only the combinations of coding rates and modulation schemes that contribute to the envelope are reported (except for the convolutional case), for the sake of pictures' clarity. In the RS case we used shortened codes which derive from the RS(255, 223) code. The erasure code relative to 23-byte blocks outperforms the 46-byte one if it is used alone (see Fig. 3). When the code is concatenated with the convolutional one, the 46-byte code slightly outperforms the 23-byte one without interleaving. We chose to represent both concatenated codes without interleaving; the reason for that is justified by the following considerations. As shown in Fig. 2, the interleaver allows a gain of about one dB in a wide range of SER values. As far as the TCP goodput is concerned, the interleaver introduces an additional latency in the order of d/μ, being d the segment depth of the interleaver. The interleaving gain is thus lower than the one evidenced in Fig. 2 [6]; furthermore, the interleaver increases the system complexity. In order to make comparisons easier, in Fig. 5 we reported the envelope of all cases shown in Fig. 4 plus the RS(239, 223) concatenated with convolutional codes. We note that, as expected, the best performance is achieved by using concatenated codes, in particular by using RS plus convolutional codes. RS code outperforms the convolutional one only at high C/N_0 values, while its performance is worse at low C/N_0 values. Anyway, we underline that the implementation of the convolutional/Viterbi technique is simpler than RS and does not have any constraints on the packet length. Erasure codes exhibit the worst performance; their most significant peculiarity is that a number of redundant blocks in a packet can not only be corrupted but may even be lost, still allowing a successful segment delivery, and an AWGN channel does not take sufficient advantage of this. On a channel that causes very long bursts of errors (in the order of the block length), even if an error burst corrupts an entire block or more, erasure codes may recover the corrupted packet, while it is difficult to do by using the other codes.

C. Multiple TCP Connections per Link - Simulation

When multiple TCP connections share the same link, it was empirically observed in [16] (by making use of simulation) that, if all connections have the same latency, they obtain an equal share of the link bandwidth. This is what we observe in our simulations as well (see Table II). We consider the case in which all links have the same latency. By using the same point-by-point optimization method, we made a comparison between the goodput achieved by a single connection with the aggregated goodput of eight TCP connections sharing the same link and the same buffer. The comparison of the envelope lines, which express the maximum values of the goodput versus C/N_0 for different coding types, is reported in Fig. 6. The whole channel bandwidth is 4 or 2Mb/s for

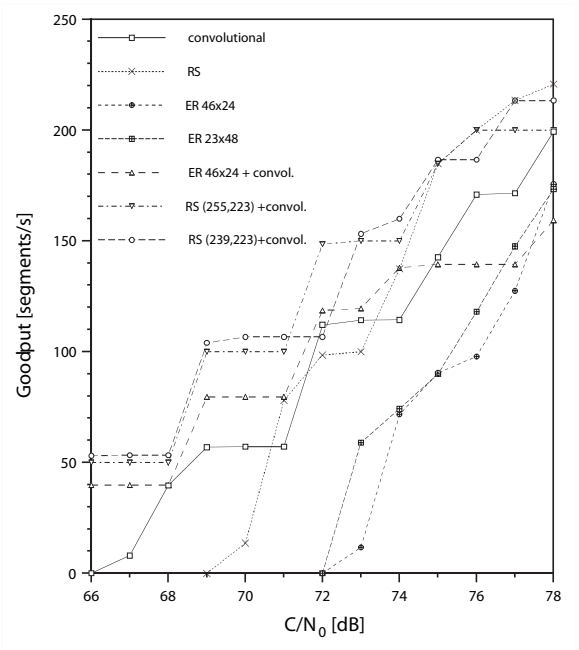

Fig. 5. Maximum goodput versus C/N_0, for different coding types (RS = Reed Solomon, ER = erasure). Each curve is the envelope of results reported in Fig. 4. The RS(239,223)+convol. case has been added as well.

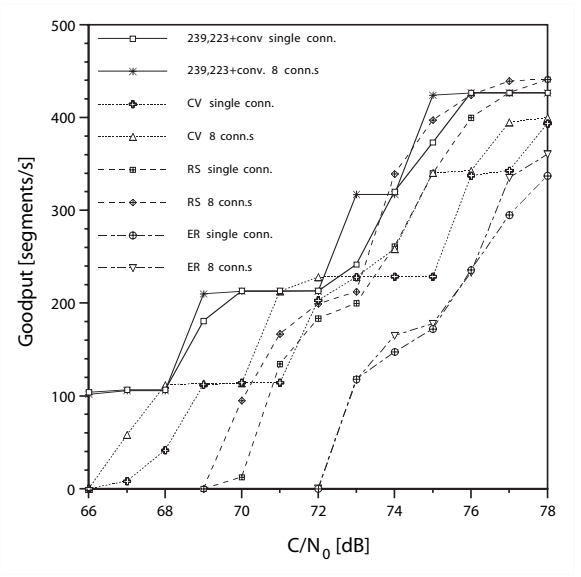

Fig. 6. Comparison between the maximum goodput of a single TCP connection and the aggregated goodput of 8 TCP connections sharing the same link and buffer, versus C/N_0, for different coding types. The RS (Reed Solomon) cases refer to codes as in Fig. 4b, while the erasure (ER) cases refer to codes as in Fig 4c. CV denotes the convolutional/Viterbi case.

QPSK or BPSK modulation schemes, respectively. Figure 6 shows that the total goodput differs noticeably in the two cases, for many of the C/N_0 values considered and for all coding types. The difference is null when the selection of the transmission parameters only allows values of the SER that are much lower than the necessary ones. We also simulated the case in which there are eight TCP connections, each of which has its own buffer and a separated link with a capacity equal to 1/8 of the entire bandwidth, i.e. 512 (256) kb/s in QPSK (BPSK) mode. The results obtained for the aggregated goodput of all connections do not substantially differ from the one obtained by considering eight connections sharing the same link and buffer (reported in Fig. 6), provided that the buffer is sufficiently large (e.g. $\beta \geq$0.8). This is a useful issue because it allows reducing the number of look-up tables necessary to take the optimum transmission parameters. In fact, different tables for a different number of TCP connections are not required. It is sufficient to divide the bandwidth allocated by the number of connections and use the resulting value to enter the table that gives the optimum transmission parameters as a function of the bandwidth in the current channel condition (C/N_0).

V. Conclusion

A fast simulation tool (TGEP) was developed to evaluate the TCP goodput for one or more connections that share a link with random and independent segment losses. The program is modifiable to entail different loss distributions. We then depicted a scenario that shows the maximum achievable TCP goodput in a realistic situation, in which a bandwidth portion of a geo-satellite transponder is assigned to a user (e.g., in TDMA mode), for one or more TCP connections. Several coding techniques, used for two different modulation schemes (B/QPSK), have been applied to packets of TCP connections, in a wide range of the signal to noise ratio of the channel, which is supposed to be corrupted by AWGN. The transmission mode of data packets is chosen in the earth station, according to the estimated signal to noise ratio and the assigned value of the bandwidth, by making use of pre-computed look-up tables. A better optimization is possible if the number of TCP connections sharing the link is known (see Fig. 6), thus allowing a further gain (over 1 dB in some cases). However, this would require a sort of cross-layer interaction, by inspecting network packets, which is not always possible, e.g. when packets carry an encrypted payload [6]. The comparison of the different coding techniques shows the superiority of the convolutional/Viterbi code, concatenated with RS codes, over the other ones. This is the result in an AWGN channel. However, we think that the strong peculiarity of erasure codes, which allow the recovery of a packet even with missed blocks, can be better exploited when different types of impairment affect the channel. Furthermore, software implementations of erasure codes allow improvements of existing systems.

The evolution of this work will be the analysis of various codes, when both long and short-lived connections coexist, and for different TCP versions (e.g. Vegas or Westwood), other than the analysis in the presence of long bursts of errors.

Appendix I

Let us denote by s_i the integer part of the division ebl/m, and by s_r the remainder of such a division. If the probability of error p is small, we can neglect the probability that the distance between two error bursts is smaller than m [25]. We can also assume that the probability of a burst of errors is equal to $p/\overline{ebl}$. For each event of burst of errors that occurs, the average number of corrupted symbols is $E\{(s_i+1)(m-s_r+1)+((s_i+2)(s_r-1)\}$. We thus have $p_s = p/\overline{ebl}\ E\{s_i m + s_r + m - 1\}$, from which relation (17) derives.

ACKNOWLEDGMENT

The author wishes to thank F. Potortì for his precious help.

REFERENCES

[1] J. S. Baras, V.G. Bharadwaj, and N. P. Butts, "An architecture for Internet service via broadband satellite networks," *Int. J. Satell. Commun.*, vol. 19, pp. 29–50, Jan./Feb. 2001.

[2] C. Partridge and T. J. Shepard. "TCP/IP performance over satellite links," *IEEE Network*, vol. 11, pp. 44–49, Sept.–Oct. 1997.

[3] M. Marchese. "Performance analysis of the TCP behavior in a geo satellite environment,' *Comput. Commun. J.*, vol. 24, pp. 877–888, May 2001.

[4] M. Allman *et al.*, "Ongoing TCP research related to satellites," RFC 2760, Feb. 2000.

[5] S. Dawkinks *et al.*, "End-to-End performance Implications of Links with Errors," RFC 3155, Aug. 2001.

[6] G. Fairhurst and L. Wood, "Advice to link designers on link Automatic Repeat reQuest (ARQ)," RFC 3366, Aug. 2002.

[7] M. Allman, D. Glover, and L. Sanchez, "Enhancing TCP Over Satellite Channels using Standard Mechanisms," RFC 2488, Jan. 1999.

[8] J. Border *et al.*, "Performance enhancing proxies intended to mitigate link-related degradations," RFC 3135, June 2001.

[9] A. Chockalingam, M. Zorzi, and V. Tralli, "Wireless TCP performance with link layer FEC/ARQ," in *Proc. ICC '99*, pp. 1212–16, June 1999.

[10] C. Barakat and E. Altman, "Bandwidth tradeoff between TCP and link-level FEC," *Computer Networks*, vol. 39, pp. 133–150, June 2002.

[11] H. Balakrishnan and R. H. Katz, "Explicit Loss Notification and Wireless Web Performance," in *Proc. IEEE Globecom Internet miniconf.*, Sydney, AU, Nov. 1998.

[12] M. Gerla *et al.*, "TCP Westwood: congestion control using bandwidth estimation," in *Proc. IEEE Globecom*, S. Antonio, TX, Nov. 2001.

[13] I. F. Akyildiz, G. Morabito, and S. Palazzo, "TCP-Peach: a new congestion control scheme for satellite IP networks," *IEEE/ACM Trans. Networking*, vol. 9, pp. 307–321, June 2001.

[14] N. Celandroni and F. Potortì, "Maximising single connection TCP goodput by trading bandwidth for BER," *Int. J. Commun. Systems*, vol. 16, pp. 63–79, Feb. 2003.

[15] E. N. Gilbert, "Capacity of a Burst noise Channel," *Bell Systems Technical J.*, vol. 39, pp. 1253–1266, 1960.

[16] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. Networking*, vol. 5, pp. 336–350, June 1997.

[17] M. Allman, V. Paxson and W. S. Stevens, "TCP congestion control," RFC 2581, Apr. 1999.

[18] M. Mathis *et al.*, "TCP selective acknowledgement options," RFC 2018, Oct. 1996.

[19] V. Jacobson, R. Braden, and D. Borman, "TCP extensions for high performance," RFC 1323, May 1992.

[20] S. Shenker and L. Zhang, "Some Observations on the Dynamics of a Congestion Control Algorithm," *ACM Comput. Commun. Rev.*, vol. 20, pp. 30–39, Oct. 1990.

[21] N. Celandroni, "TCP Goodput Evaluation Program," [Online], Available: http://wnet.isti.cnr.it/software/tgep.html, 2004.

[22] J. Padhye *et al.*, "Modeling TCP Reno performance: a simple model and its empirical validation," *IEEE/ACM Trans. Networking*, vol. 8, pp. 133–145, Apr. 2000.

[23] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, pp. 379–413, Aug. 1993.

[24] "Q1401: K=7 rate 1/2 single-chip Viterbi decoder technical data sheet," *Qualcomm incorporated*, Sep. 1987.

[25] M. K. Simon *et al.*, "Spread Spectrum Communications Handbook," rev. ed.: McGraw-Hill, 1994.

[26] M. Riley and I. Richardson, "Reed Solomon Codes," [Online]. Available: http://www.4i2i.com/reed_solomon_codes.htm, Nov. 2002.

[27] R. Blahut, "Theory and Practice of Error Control Codes," Reading, MA: Addison-Wesley, 1983.

[28] S. Lin and D. J. Costello, *Error Control Coding: Fundamental and Applications*, Prentice-Hall, 1983.

[29] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM Comput. Commun. Rev.*, vol. 27, pp. 24–36, Apr. 1997.

Nedo Celandroni received the Dr. Ing. degree in Electronic Engineering from the University of Pisa, Italy, in 1973. Since 1976 he has been a researcher with the CNUCE Institute (now ISTI) of the Italian National Research Council (CNR). He worked for the realization of the Flight Dynamic System of the SIRIO satellite project. Since 1979 he has been involved in the field of digital satellite communications. He participated in several projects in this field: STELLA I/II (Satellite Transmission Experiment Linking Laboratories), FODA (Fifo Ordered Demand Assignment), FODA/IBEA (Information Bit Energy Adaptive), Progetto Finalizzato Telecomunicazioni, Experiments on the satellites Olympus and Italsat. His interest includes rain fade countermeasure systems, data quality estimation, VSAT systems, GEO, MEO and LEO satellites for mobile telephony and multimedia systems, terrestrial wireless networks, TCP congestion management, and TCP over wireless channels. He obtained two patents, in 1989 and 1996, for the design of the FODA and FODA/IBEA systems, respectively. At present he is involved in the European project (Network of Excellence) SatNEx and in the Italian project IS-MANET.