IETE Journal of Research
Vol 52, Nos 2 & 3, March-June, 2006, pp 121-137

Modifications of the Slow Start Algorithm to Improve TCP Performance Over Large Delay Satellite Channels

MARIO MARCHESE

Department of Communication, Computer and System Science (DIST) / CNIT (Italian National Consortium for Telecommunications), University of Genoa, via Opera Pia 13, 16145 Genova (Italy).
e-mail: mario.marchese@cnit.it

The paper presents a study and performance analysis of a modified version of the TCP (Transmission Control Protocol) over a GEO (Geostationary Orbit) satellite link. The Round Trip Time (RTT) is above 500 ms. The high delay to receive acknowledgements decreases the performance of the TCP but, if the application field has a limited extension (as it is the test-bed emulated in this paper) the characteristics of the links and of the devices traversed are well known and the congestion aspects may be treated differently than in a large cable network, where the number of devices in the path is difficult to control. So, even if the performance of TCP is not satisfying, improvements can be obtained by properly tuning some parameters and modifying algorithms.

A new increase function of the slow start algorithm is introduced in the paper. It is called MTSI - Multi Threshold Smoothed Increase. The performance is measured by the throughput in bytes/s and by the overall transmission time. An ftp-like application designed for the aim has represented the reference application. The performance analysis is carried out by using both a real satellite test-bed and a satellite network emulator. The new proposal is compared with different strategies already in the literature, including the TCP New Reno, configurations with augmented buffer and initial window and transport layer solutions already adapted for the satellite environment. The analysis has included both the single and the multiple application case, where several connections share the satellite link at the same time. The file transfer dimension is varied along with the bandwidth available and the advantage of the new proposal depending on these two parameters is also evidenced. The performance has been measured both in clear sky condition (no packet loss) and during loss situations.

Indexing terms: Satellite communications, Network protocols; TCP/IP, TCP over satellite.

1. INTRODUCTION

SATELLITES offer clear advantages with respect to cable networks [1]. The architecture is scalable: a new user can join a satellite communication by acquiring the necessary technical instrument, no area shall be wired to get the high-speed service. The diffusion throughout the land is wide: a satellite network overcomes simply the geographical obstacles, which will make difficult the installation of a cable network of equivalent quality; moreover, satellites can cover isolated areas. The bandwidth availability is high: Ka-band (20-30 GHz), which is the object of many experiments and also of the tests over the real network reported in this paper, is less affected by congestion than terrestrial networks, where, even if the bandwidth may be broad, the number of potential users is huge. The multicast service is very simple, being the satellite inherently a broadcasting tool. Finally, satellite links are often private lines, unlike submarine and overland networks: a private network has the advantage of being managed by few people so to avoid many problems about the property and organisation of different network portions.

As a consequence, the interest in delivering TCP/IP services over satellite is clear. Unfortunately, the network used heavily affects the behaviour of the protocols; even in a satellite environment, the problems are different if a LEO (Low Earth Orbit), a MEO (Medium Earth Orbit) or a GEO (Geostationary Orbit) satellite system is used [2]. Issues related to each environment are listed in [3]. Of particular importance in this environment is the performance of the transport protocol (TCP, in the treated scenario). TCP, even if it still works, is not efficient over channels characterized either by large delay-bandwidth product or by losses due to channel errors. The large product (as in geostationary satellite links) makes the acknowledgement scheme on which TCP is based, very inefficient. On the other hand, TCP considers any loss as a congestion event and decreases the bit rate entering the network. If a loss is due to a channel event (e.g. rain fading), reducing the rate does not bring any advantage. That is the typical case of LEO satellites and radio networks.

The problem of improving the transport layer over

Paper No 136-B; Copyright © 2006 by the IETE.

satellite has been investigated in the literature for some years: reference [4] is one of the first overview on the topic and reference [5] is a more specific study in TCP/IP networks with high delay per bandwidth product and random loss. More recently, reference [6] provides a summary about improved TCP versions and lists issues and challenges in satellite TCP as well as possible enhancements at the link layer; the work in [7] reports the main limitations of the TCP over satellite and proposes many possible methods to act. Reference [8] represents a complete tutorial paper on the topic: various possible improvements both at the transport level and at the application and network level are summarized and referenced; the paper focuses also on large delay per bandwidth product network and suggests possible modifications to TCP, as the buffer size. A recent issue of International Journal of Satellite Communications is entirely dedicated to IP over satellite [9].

In more detail, the approaches in the literature may be structured into two broad categories (partially from [10]): "Pure transport layer", where the modifications are implemented at the transport layer of the terminal hosts without any intervention in the middle of the end-to-end path and "Hard state transport layer", where all forms of transport layer splitting are allowed to improve the performance.

Many works may be classified within the "Pure transport" categories: all the different TCP versions as TCP Tahoe [11], Reno[12], Vegas [13, 14], NewReno [15], TCP SACK [16] and the extension proposed in reference [17], which allows a more effective recovery mechanism by exploiting the advantages provided by a selective repeat scheme; all the proposals concerning TCP parameter setting and tuning as in reference [18], which shows an analysis of the TCP behaviour by varying parameters as the buffer size and the initial congestion window on the basis of previous extensive work about the topic [19-21]; TCP Eifel [22] that allows specifying a better timeout management to avoid spurious timeout and useless retransmissions; TCP Westwood [23] and Westwood+ ([24], taken also as a comparison in this work) and TCP-Veno [25], which employs a rate-based transmission algorithm inherited from TCP-Vegas implementation and specifies a more effective recovery mechanism able to estimate the available channel bandwidth without reducing the congestion window too drastically; TCP Peach [26] and Peach+ [27] that implement a recovery mechanism more effective than TCP NewReno by exploiting the knowledge carried in probing dummy (Peach) or NIL (Peach+) segments and able to assure satisfying results also when the channel state is very critical in terms of packet error rate.

Concerning "Hard-state transport": the concept that a satellite network portion may be isolated and receive a different treatment and attention with respect to the cabled parts of the network is much investigated in the literature. Methodologies as TCP splitting [4,6,28], "where a TCP connection is divided into multiple TCP connections, with a special connection running over the satellite link" [29] and TCP spoofing [4], where "a router (gateway) near the source sends back acknowledgements for TCP segments in order to give the source the illusion of a short delay path and therefore to speed up the sender's data transmission" [29], bypass the concept of end-to-end service with the aim of isolating the satellite link. Recent RFC 3135 [30] is dedicated to extend this concept by introducing Performance Enhancing Proxies (PEPs) intended to mitigate link-related degradations. RFC 3135 is a survey of PEP techniques, not specifically dedicated to the transport layer, even if emphasis is put on it. Motivations for their development are described as well as consequences and drawbacks. There are also commercial products implementing TCP-PEP. Reference [29] contains some examples. Often specific transport layers are designed to improve the performance in this framework: the proposal in reference [31], for instance, where the ways in which latency and asymmetry impair TCP performance are also investigated.

Also International Standardization Groups as the Consultative Committee for Space Data Systems - CCSDS and the European Telecommunications Standards Institute - ETSI, which is running its activity within the framework of the SES BSM (Satellite Earth Station - Broadband Satellite Multimedia) working group, are active on TCP over satellite issues. Of particular relevance are references [32] and [33].

The present paper focuses on a GEO system with a large delay-per-bandwidth-product and symmetric channel.

The paper focuses on the effect of modifying the slow start algorithm, after tuning the initial congestion window and the buffer length. The new algorithm, called MTSI (Multi Threshold Smoothed Increase) is based on the introduction of a different slow start increasing function F(·), which depends on the number of received acknowledgements and on the current dimension of the congestion window. The work analyses the protocol behaviour and its performance. Even if the proposal concerns a strict modification of a TCP algorithm (the slow start), it may be implemented directly on the operating systems of the terminal hosts (actually the performance analysis has been carried out in this way) and be considered also a "Pure Transport Layer", it has not a general scope. It must be applied only over a particular satellite environment. It means that it can be regarded as a transport layer implementation for the satellite portion of the network and its application framework is "Hard state transport layer".

Being an experimental work, the tests are performed on a particular network and the numerical results strictly depends on the type of network. Nevertheless, the methodology of design introduced is not affected. A TCP New Reno with Selective Acknowledgement (SACK) mechanism [16], implemented over Linux kernel 2.2, is utilised to implement all the TCP versions presented in the paper (except for "TCP Westwood+", whose software has been designed for Linux kernel 2.4) and to get the results in the paper. The performance analysis has been carried out over a real test-bed and over a satellite emulator. The paper is structured as follows. Section 2 contains the MTSI slow start algorithm proposed. Section 3 describes: the operative scope of the research, the test-bed network, the application used to get the results and the emulation tool utilised to obtain part of the measures. The performance analysis is contained in section 4. Section 5 presents the conclusions.

2. THE SLOW START ALGORITHM PROPOSED

2.1. Slow start main characteristics

The short summary in the following is intended to focus on the TCP characteristics that concern the paper. The parameters are substantially set by following the standard proposed in [12] and [15]. The notation used herein has been introduced in [12].

The slow start phase begins setting:

- the congestion window (cwnd) to 1 segment (1·smss, where smss, measured in bytes, stands for Sender Maximum Segment Size); more exactly, "IW, the initial value of cwnd, must be less than or equal to 2·smss bytes and must not be more than 2 segments" [12].
- the slow start threshold (ssthresh) to a very high value (infinite).

At each received acknowledgement (ack), cwnd is increased by 1·smss (i.e. "ACK $\rightarrow$ cwnd = cwnd + 1 · smss").

The real transmission window TW is set, TW = min{cwnd, min(source buff, rwnd)}, which is the minimum between cwnd and the minimum between the source buffer and rwnd, where rwnd is the most recently advertised receiver window (a receiver-side limit on the amount of outstanding data [12]). It corresponds to half of the receiver buffer length, at the beginning of the transmission, in Linux kernel 2.2 implementations, and to the receiver buffer length in Linux kernel 2.4 software.

2.2. The modified version

A parameterisation of the initial congestion window (IW) for satellite links is followed in the paper by setting cwnd = IW· smss. TCP algorithms as congestion avoidance, fast retransmit and fast recovery are untouched.

The paper proposes a parameterisation, called MTSI (which stands for Multi Threshold Smoothed Increase), of the slow start increase function " ACK $\rightarrow$ cwnd = cwnd + 1· smss". The algorithm, defined below, is based on the introduction of a different increasing function $F(\cdot)$, whose value when the Nth acknowledgement (Ack_N) is received depends on the current dimension of the congestion window (cwnd) and on the value of F(·) when the (N − 1)th acknowledgement is received (Ack_{N-1}).

In short,

$$\mathrm{cwnd}(\mathrm{Ack}_N) = \mathrm{cwnd}(\mathrm{Ack}_{N-1}) + F(\mathrm{Ack}_{N,}\ \mathrm{cwnd}(\mathrm{Ack}_{N-1})\cdot \mathrm{smss} \tag{1}$$

The function F(·), introduced in [34] is aimed at regulating the size of the congestion window in the slow start phase. The characteristics of F(·) affect the increase of the window and, as a consequence, the transmission speed and the protocol performance. The definition of F(·) is not trivial and many considerations may affect the decision: the choice performed in this paper is aimed at increasing the transmission speed in the initial phase on the basis of the experiments performed with an augmented value of IW without entering a congestion period. The function introduced in the paper is called $F_{MTSI}(\cdot)$. The choice allows tuning the behaviour of the protocol depending on the congestion window, and to measure, at some extent, the network status represented by the arriving acknowledgements.

TCP sets the function

$$F(\mathrm{Ack}_{N,}\ \mathrm{cwnd}(\mathrm{Ack}_{N-1})) = 1 \tag{2}$$

$F_{MTSI}(\cdot)$ sets the function as in formula (3), where the increase is smoothed over time, depending on the threshold thr, linked to the dimension of the congestion window and defined in formula (4).

$$F_{MTSI}(\mathrm{Ack}_N \mathrm{cwnd}(\mathrm{Ack}_{N-1})) = \begin{cases} F_{MTSI}(\mathrm{Ack}_{N-1,}\ \mathrm{cwnd}(\mathrm{Ack}_{N-2})) + K_{thr}(\mathrm{Ack}_N) & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_n \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

$$\mathrm{Kthr}(\mathrm{Ack}_N) = \begin{cases} K_{thr_1} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_1 \\ K_{thr_2} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_2 \\ K_{thr_3} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_3 \\ \cdot\ \cdot & \\ \cdot\ \cdot & \\ K_{thr_n} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_n \end{cases} \tag{4}$$

A variable number of thresholds (i.e. thr_n, where $n \in N$) may be used. $F_{MTSI}(.)$ is aimed at adapting the protocol behaviour through the constants (K_{thr_n}, if n thresholds are used). Three thresholds have been heuristically estimated to be a proper number to increase the rate in the first phase of the transmission and to smooth it over time. The function chosen appears as in (5).

$$K_{thr}(Ack_N) = \begin{cases} K_{thr_1} & \text{if cwnd}(Ack_{N-1}) < thr_1 \\ K_{thr_2} & \text{if cwnd}(Ack_{N-1}) < thr_2 \\ K_{thr_3} & \text{if cwnd}(Ack_{N-1}) < thr_n \end{cases} \quad (5)$$

The notation used to specify the threshold involved is MTSI ($thr_1 - thr_2 - thr_3$); the value of the constant K_{thr_n}, with $n \in \{1,2,3\}$, represents the angular coefficient of the linear increase. Its value governs the speed of the increase and rules the protocol behaviour. The values of the constants will be identified in section 4, dedicated to the results.

2.3. Alternatives functions F(·) for comparison

The behaviour of the new algorithm has been compared with other types of functions F(·), characterised by a more aggressive slope in the first part of the transmission and a smoothed behaviour after a threshold (see below for the definition of the individual functions already investigated in [34]).

Concerning the first one: at each acknowledgement received, the increment is constant (equal to the quantity K) and greater than 1. The first function proposed is not dependent on the number of received acknowledgements and on the congestion window. The function F(·) is set as in formula (6). CONST stands for constant.

$$F_{CONST}(Ack_N, \text{cwnd}(Ack_{N-1})) = K, K > 1 \quad (6)$$

In the second alternative F(·) as in formula (7)), the increment is linear up to the value "thr" of a fixed threshold; it is constant after this value. This method is referenced as "Linear thr" in the results presented (i.e., if thr = 20, the method is identified as "Linear 20").

$$F_{MTSI}(Ack_N, \text{cwnd}(Ack_{N-1})) = \begin{cases} F_{MTSI}(Ack_{N-1}, \text{cwnd}(Ack_{N-2})) + 1 & \text{if cwnd}(Ack_{N-1}) < thr \\ 1 & \text{otherwise} \end{cases} \quad (7)$$

3. OPERATIVE SCOPE

3.1. Scope of the work

The new algorithm may be applied both directly on the terminal hosts without any intervention on the network tools and also used within a network including non-satellite portions by isolating the satellite network from the rest and applying the solution only to the satellite portion. Even if the performance analysis has been obtained by applying the first choice, the suitable application field of the proposal is a TCP splitting - PEP architecture. Figure 1 contains a graphical representation of this environment. TCP acts over the non-satellite portions of the network while the new proposal (identified here as "TCP MTSI", coherently with the function $F_{MTSI}(\cdot)$, defined in the previous section, applies only over the satellite portion.

Even if this work concerns only transport layer, all the protocol stack acting on the satellite portion has been evidenced with a bolded rectangle in Fig 1 to mean that, in the future, all the stack, including lower layers may be modified. A possibility is represented by "Complete Knowledge" approach [35], where each network device is completely known and a new stack, totally adapted to the satellite link, can be designed over the gateway. It opens new possibilities to the investigation because the definition of an overall stack includes the optimisation of Layer 3 (IP), Layer 2 (LLC and MAC) and Resource Allocation (which is topical for this solution). It should offer a performance upper bound even if the implementation could be complex. It may be the object of future study.

3.2. Real test-bed and application

The real test-bed where the new algorithm is applied to get part of the results appearing in this work is shown in Fig 2. "TCP MTSI" is directly implemented in the operating systems of the hosts to simplify the experiments. Only two terminal PCs (within Local Area Networks – LANs) have been shown because a TCP connection concerns a couple of processes (source and destination). They are connected through a satellite link by using two IP routers through a 10 Mbits/s link. The TCP/IP protocol stack is used. The data link level of the router uses HDLC encapsulation on the satellite side, where a serial interface is utilised, and Ethernet on the LAN side. A raw Bit Error Rate - BER (i.e., BER with no channel coding) approximately of 10^{-2} has been measured; the utilisation of a sequential channel coding with a code rate of 1/2, to correct transmission errors, has allowed to reach a BER of about 10^{-8}.

The system employs the ITALSAT II satellite, providing a countrywide coverage in the Ka-band (20-30 GHz), which is currently explored for the provision of new services. The overall bandwidth is 36 MHz. Each satellite station can be assigned a full-duplex dedicated traffic channel with a bit-rate ranging from 32 Kbits/s to 2 Mbits/s (this latter adopted for the tests) and it is made up of the following components:

- Satellite Modem, 32 Kbits/s – 2 Mbits/s satellite modem, connected to the RF Device.
- RF Device, 1.8 meters antenna aiming at ITALSAT II.
- IP Router connected to:
- Satellite modem via RS449 Serial Interface
- Application Host via Ethernet IEEE 802.3 10BASE-T link
- Application PC. It is the source of the TCP/IP traffic under test, implemented over the operating system Linux (kernel 2.2).

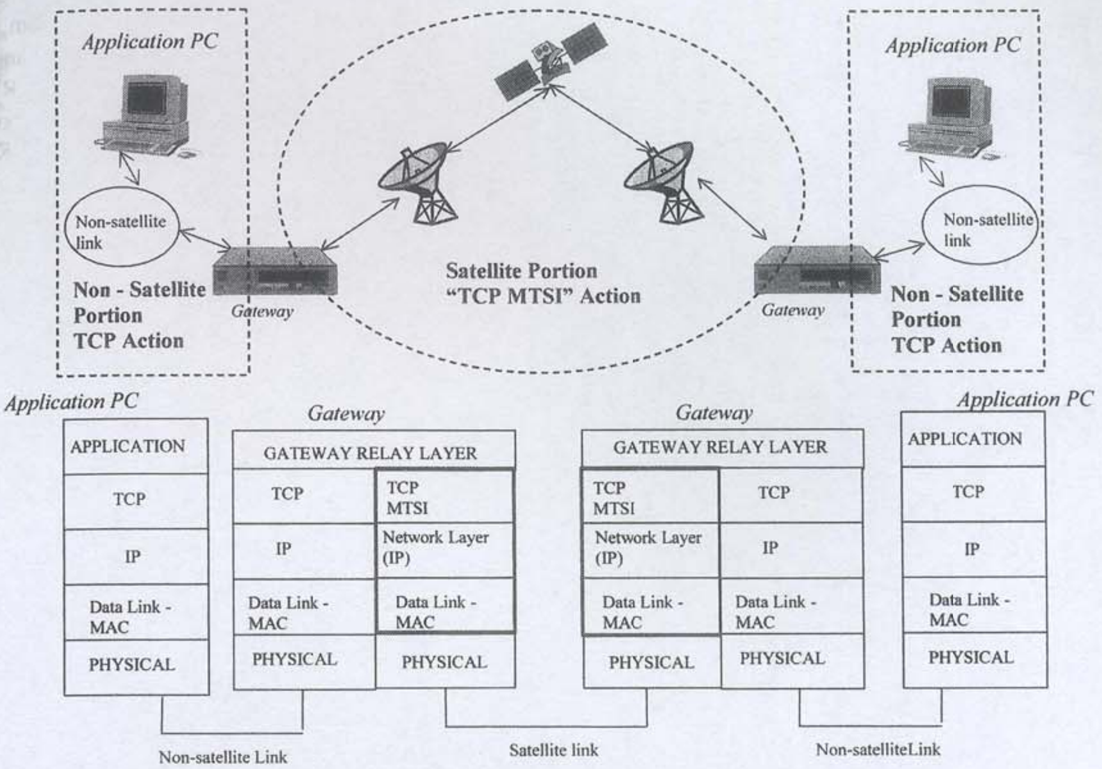

Fig 1 TCP splitting architecture

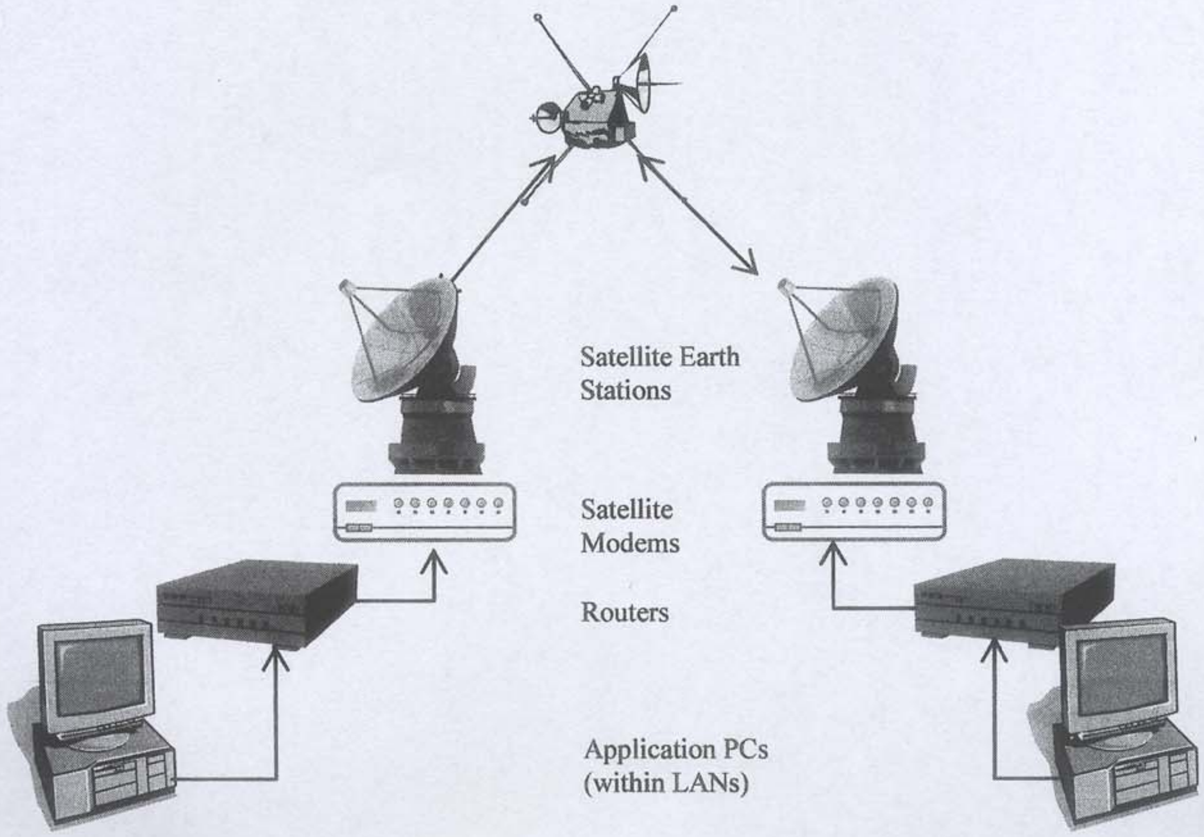

Fig 2 Test-bed network

The application used to get the results is a simple ftp-like one, i.e., a file transfer application located just above the TCP. It allows transferring data of variable dimension between the two remote sites; the work has taken this application as a reference because it is thought as fundamental for most of the applications of interest. The modified version of the TCP is located in the Application PCs.

In general, the configuration of the buffers within the intermediate router would deserve a particular attention. Actually the software in the router along with the queues' dimension and FIFO (First In First Out) management policy has not been modified to perform the tests over the real test-bed but some more detail may help understand. A TCP connection may be roughly modelled as in Fig 3, which contains only the essential elements for the following discussion, from the memory point of view. The TCP buffer is served at the channel speed (10 Mbits/s, in the Ethernet of the test-network mentioned), when there are some segments to serve, and the outgoing traffic enters the IP buffer of the router. The server of the IP router buffer, in the model, is the satellite modem, which works at 2 Mbits/s (when there are some data). The formal notation (I(t), O(t), O'(t)) has been introduced to take into account the periods of inactivity. A certain amount of data may be kept in flight inside the pipe represented by the satellite channel: the capacity of the pipe is given by the product (Round Trip Time) × (bandwidth available); e.g. if the mentioned GEO system is used: RTT (0.511 s) × Bandwidth (2 Mbits/s)). Actually, the memory defined includes the segments in flight plus the segments already at the destination but not yet acknowledged at the source. If there is at least one segment in the IP buffer and one segment at the destination, then $O'(t) = O''(t)$. So, except for the initial phase, when $O'(t)$ works and $O''(t) = 0$, and the final phase of the connection, when the opposite situation happens, $O''(t)$ should be equal to $O'(t)$ for most part of the connection time.

Even if rough, the model allows also a deeper comprehension of the overall behaviour and of the results. For example, the value of the TCP buffer length 320 Kbytes (equally set at the source and destination), which allows a rwnd of 160 Kbytes corresponding to half of the receiver buffer length at the beginning of the transmission (as specified in section 2.1), obtained from the analysis in [18] as the configuration providing the best results, apparently overcomes the theoretical capacity of the system that is given by the product RTT (Round Trip Time) per bandwidth (the satellite "pipe"). In the case shown RTT is approximately 511 ms and the bandwidth corresponds to 2 Mbits/s. The product provides approximately 130 Kbytes. The value 160 Kbytes, which is the transmission window bottleneck because it is half of the receiver buffer length, is greater than 130 Kbytes. The observation ignores the memorization capacity of the other components of the system, as the IP buffer. For example, it is more than sufficient to have an IP buffer of 40 full-sized packets of 1500 bytes each (60 Kbytes globally, as actually is in the two routers used to get the results in the real test-bed) to enlarge the system capacity without causing any packet loss in the intermediate router.

In general, the choice of the buffer dimension, in particular the IP buffer length, which is fixed in the approach taken in this work, and the dynamic of the segment arrivals (I(t)) is important. An excessive increase of the IP buffer length provides unpleasant effects, due to the Retransmission Timeout (RTO) of the TCP. A RTO time after sending a segment, the TCP sets the Initial Window to 1·smss (IW·smss, if the dimension of the initial window is parameterised), where smss stands for Sender Maximum Segment Size, and begins again the transmission by using the Slow Start algorithm. This situation is represented in Fig 4, where one transfer of about 2.8 Mbytes (2788848 bytes, exactly) file is shown by using two different configurations for the transport layer and the IP buffer. In more detail, the function $F_{CONST}(\cdot)$, defined in the previous section, is set to a constant value $K = 10$; the initial window IW is set to 6. The configuration setting a TCP buffer of 320 Kbytes and an IP buffer in the router of 60 Kbytes (40 segments of 1500 bytes) has been compared with a 10 Mbytes (TCP buffer) – 6.144 Mbytes (4096 segments, IP buffer) configuration. The configurations have been implemented in the test-bed described above and the results reported again really measured in the field. The throughput overcomes the channel bandwidth of 2 Mbits/s because it is strictly referred to the information source (the server): the aim is to focus only on the saturation. A huge buffer, as in the (10 – 6.144 Mbytes) case, should guarantee a non- loss behaviour. The configuration has a steep initial increase, due to the function $F_{CONST}(\cdot) = K = 10$. The throughput reaches 600 Kbytes/s after about 4 seconds but, after this phase, the performance dramatically decreases. The

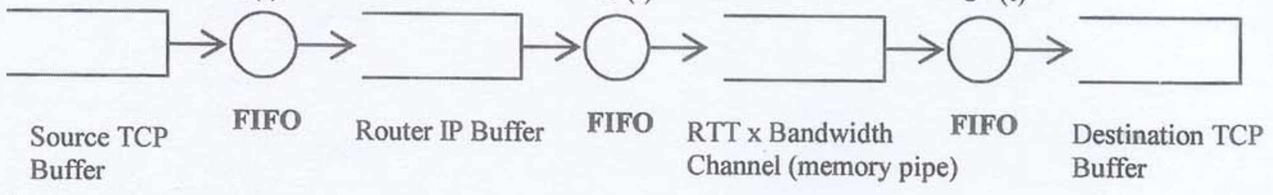

Fig 3 Memorization model for a TCP connection.

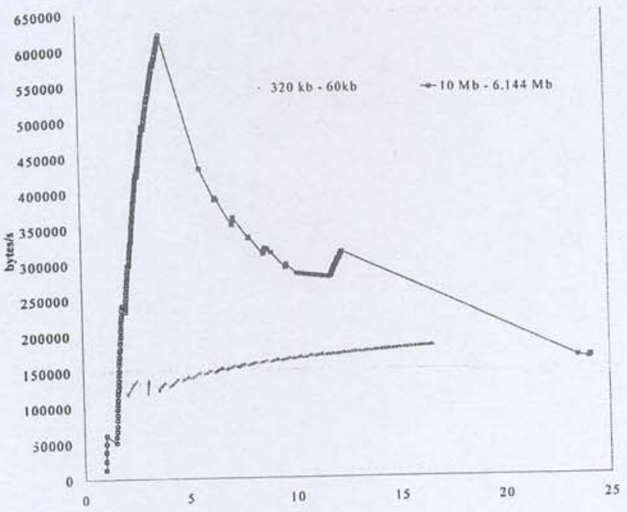

Fig 4 Throughput vs time, H = 2.8 Mbytes, K = 10, IW = 6, TCP-IP buf

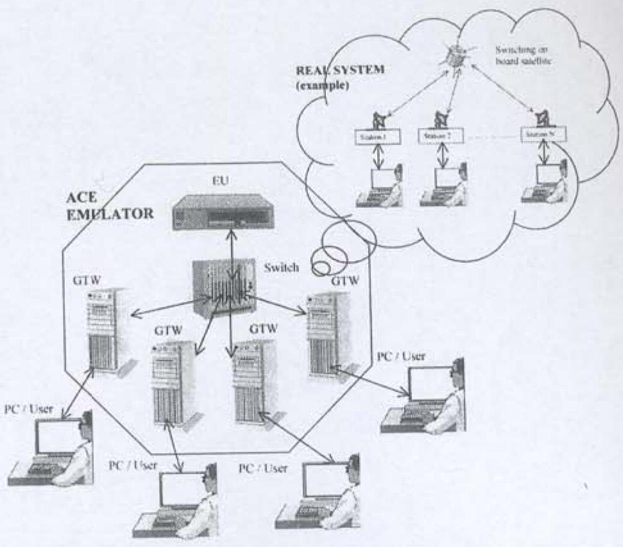

Fig 5 Emulator architecture

motivation is clear by analysing the traces obtained. A short summary about them is reported in the following: after 4 seconds, the IP buffer contains about 1800 Kbytes (1200 segments). The transmission goes on but the last segments entering the IP buffer stay queued for a long time. After a time RTO, the congestion window is strongly reduced. It is set "to no more than the loss window, LW, which equals 1 full-sized segment (regardless of the value of IW). Therefore, after retransmitting the dropped segment the TCP sender uses the slow start algorithm to increase the window from 1 full-sized segment" [12]. So, segments already sent, which are not lost but are waiting the service in the IP buffer, are uselessly re-transmitted. The overall performance (e.g., the throughput and the overall transmission time) drastically decreases.

It is important to remember that the TCP buffer length has been equally set both at the source and at the destination. The test reported in Fig 4 has been obtained with only a connection in the network (mono- connection case).

3.3. Emulator

The satellite test-bed network has been also real-time simulated by using an emulation tool, developed in a similar context [36], properly tuned for the aim and briefly described below. It is possible to identify the following main parts: a modem with an interface towards the upper layers (namely the network layer); a channel characterized by its own peculiarities and a data link protocol over the satellite channel. The emulation tool can describe the behaviour of each packet in the satellite network at the layer 2, including a switching on-board architecture, if necessary.

The emulator reference architecture is shown in Fig 5, along with one possible system to be described enclosed in the cloud (a GEO satellite system with N earth stations has been depicted in this case, N = 2 is the configuration used in the tests). Different units called Gateways (GTWs) operate as interface among the emulator and the external PCs. Each GTW is composed of a PC with two network interfaces: one towards the external world (a 10/100 Mbits/s Ethernet card), the other towards the emulator. An Elaboration Unit (EU), which has a powerful processing capacity, carries out most of the emulation.

The interface towards the external world concerns the GTWs; the loss, delay and any statistics of each packet regards the EU; the real transport of the information through the network concerns the input GTW and the output GTW. The various components are connected via a 100 Mbits/s network, completely isolated by a full-duplex switch. In such way, the emulator has an available bandwidth much wider than the real system to be emulated, which should not overcome a maximum overall bandwidth of 10/20 Mbits/s.

In more detail, Fig 6 shows how the different parts of the real system (modem, data link protocol, channel and on-board switching system, not used in this work) are mapped onto the different components of the emulator. It is clear in Fig 6 that the architecture of the emulator is not exactly correspondent to the real system. The earth station, identified by the grey rectangle, is divided, in the emulator, into two parts (GTW and EU). The network layer (which implements the necessary IP router functionalities), the network interface towards the external world and the interface between the network layer and the satellite modem are contained in the Gateway (GTW). The other parts of the modem (i.e. the data link layer, protocol and encapsulation), the overall transmission characteristics (e.g. bit error rate, channel fading, lost and delayed packets), the on-board switching architecture as well as the queuing strategies are contained in the Elaboration Unit (EU). Referring to the discussion of the previous sub-section, an IP buffer of 100 full-sized packets of 1500 bytes each has been set at the transferred in the mono-connection case. The tests in the multi-connection case are obtained by activating transfers of 100 Kbytes (102808 bytes) each.

"TCP MTSI" is compared with:

- TCP (Initial Window IW = 1 and buffer set to 64 Kbytes, recommended values for TCP implementations), identified as "TCP IW=1 buf = 64 Kbytes";
- TCP using an extended buffer length (320 Kbytes) together with an extended initial window (IW = 6), identified as "TCP IW = 6 buf = 320Kbytes", extensively analysed in [18];
- "TCP CONST" (using $F_{CONST}(\cdot)$), buffer length 320 Kbytes, IW = 6;
- "TCP LINEAR" (using $F_{LINEAR}(\cdot)$), buffer length 320 Kbytes, IW = 6.

The comparison with "TCP IW = 1 buf = 64Kbytes" allows to have a global idea about the improvement guaranteed by the new increase strategy, while the comparison with TCP implementing only extended buffer and initial window allows to focus on the gain guaranteed by the slow start modification independently of the contribution given by buffer and initial window. Comparing the performance of $F_{MTSI}(\cdot)$ with $F_{CONST}(\cdot)$ and $F_{LINEAR}(\cdot)$ highlights the improvement introduced by the new strategy and helps tune the proper thresholds.

Figure 7 reports the performance (instantaneous throughput versus time) of two versions of "TCP MTSI" by using " $thr_1 = 20 - K_{thr_1} = 4$, $thr_2 = 30 - K_{thr_2} = 2$, $thr_3 = 40 - K_{thr3} = 1$" (identified as "TCP MTSI 20-30-40) and "$thr_1 = 10 - K_{thr_1} = 4$, $thr_2 = 20 - K_{thr2} = 2$, $thr_3 = 30 - K_{thr_3} = 1$" (identified as "TCP MTSI 10-20-30), compared with "TCP IW = 1 buf = 64Kbytes".

The window increase rate of the TCP is smoothed; the different TCP sender pacing, imposed by formula (3), helps avoid congestion. The gain with respect to "TCP IW = 1 buf = 64Kbytes" configuration is above 74% for both the configurations in the single connection case. It is important to observe that the advantage of using a modified configuration is really impressive, if compared with a TCP implementing recommended values for TCP buffer and IW. It has to be remembered that the gain provided is given by the contemporaneous and beneficial effect of the increased buffer length and initial window, as well as of the function MTSI. The following tests and the second part of the results will allow checking the effective contribution of $F_{MTSI}(\cdot)$, independently of buffer length and IW. A gain above 74% means that the transfer time for a file of about 2.8 Mbytes is reduced from a value of about 50 s to a value ranging from 12s to 13s. The advantage in the quality of service perceived by a user who performs, for instance, the download of an image file via satellite may be simply figured out.

The behaviour in the multiple connection case is reported in Fig 8 for the configuration with $F_{MTSI}(\cdot)$ that resulted as the most efficient in the single connection case ("TCP MTSI 20-30-40" that, in the remainder of the paper, will be identified only with "TCP MTSI"). "TCP MTSI" shows a very satisfying behaviour in comparison with "TCP IW = 1 buf = 64 Kbytes". The advantage given by the new algorithm is maintained up to a relevant number of connections in the network, considering the limited amount of bandwidth.

Table I summarises the results in the single connection case concerning the overall transfer time and average throughput comparing "TCP MTSI" with: "TCP IW = 1 buf = 64 Kbytes", "TCP IW = 6 buf = 320Kbytes", "TCP CONST" (using $F_{CONST}(\cdot)$ with K = 2 and K = 4), "TCP LINEAR" (using $F_{LINEAR}(\cdot)$ with thr = 10, thr = 20 and thr = 50).

The gain is referenced to the value of "TCP IW = 1 buf = 64Kbytes". For example, the Gain (in percentage) of "TCP MTSI" is computed as follows: T = 12.57s, T_{REF} = 49.21:

$$\text{Gain} = \frac{T_{REF} - T}{T_{REF}} \cdot 100 = \frac{49.21 - 12.57}{49.21} .100 = 74.5\%$$

The gain is really good for any configuration and it rises up to 74.5 % if the new function $F_{MTSI}(\cdot)$ is utilised.

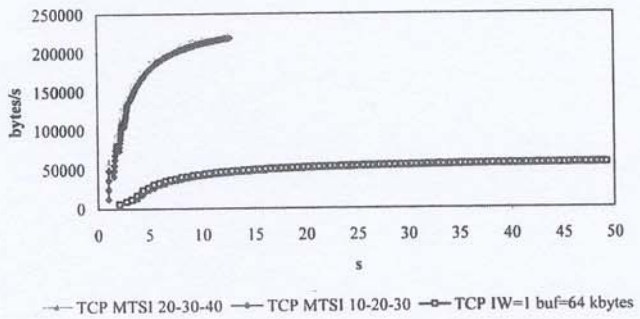

Fig 7 Instantaneous throughput (bytes/s) versus time by using function $F_{MTSI}(\cdot)$, 2.8 Mbytes

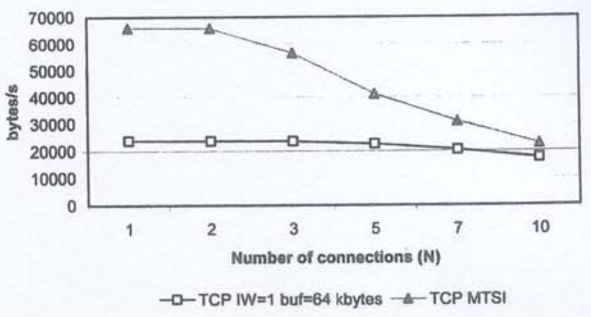

Fig 8 Throughput per connection (bytes/s) versus number of connections, $F_{MTSI}(\cdot)$, multi-connection case, 100 Kbytes

Table 2 contains transfer time, average throughput and gain provided by the configurations reported in Table 1, but the gain is referenced to "TCP IW = 6 buf = 320 Kbytes" (T_{REF} = 13.96s). It helps show the advantage given by the slow start implementation with no contribution of the initial window and buffer length. "TCP MTSI" guarantees a gain of about 10% for a file transfer of 2.8 Mbytes. The second part of the results, where the file dimension and the bandwidth is varied, will allow finding out the gain in dependence these parameters but also Fig 9, which contains the instantaneous throughput over time, for "TCP MTSI" and the configurations in Table 2 offering the best performance also in the multi - connection case [34], allows to focus on the throughput evolution over time and to have a first idea of the gain offered by "TCP MTSI" in dependence of the amount of bytes transmitted, as evidenced by the vertical grid.

The configurations providing the best results when more connections are routed have been selected and shown in Fig 10. Due to the relative extension of the bandwidth available, the use of configuration "TCP MTSI" is efficient for a limited number of connections but, at the same time, it allows avoiding congestion when the load increases. The modified configurations are substantially equivalent if the number of active connections is more than 5. These values have not been shown.

4.3. Emulation

This part of the results is obtained by using the network emulator described in section 3.3. Tests are performed by varying the bandwidth availability (500 Kbits/s, 1 Mbit/s, 2 Mbits/s) as well as the dimension of the file transfer, so that the characteristics of the new solution can be investigated in dependence of both of them. File dimensions considered are listed in the following, precise values are reported in brackets: 10 Kbytes (10136 bytes), 100 Kbytes (102808 bytes), 1 Mbyte (1048352 bytes), 3 Mbytes (3145056 bytes), 10 Mbytes (10486416 bytes).

It is divided into two sub-sections: the first one investigates a situation where there are no channel errors (ideal channel); the second one refers to a faded channel with an average packet loss of 1%.

4.3.1. *Ideal Channel*

In this case, "TCP MTSI"(which applies, again, an Initial Window of 6·smss, a source/receiver buffer length of 320 Kbytes and a function F(·) utilized with $thr_1 = 20 - K_{thr_1} = 4$, $thr_2 = 30 - K_{thr_2} = 2$, $thr_3 = 40 - K_{thr_3} = 1$) is compared with:

- TCP (Initial Window IW = 2 and buffer set to 64 Kbytes, used within the more recent operating system protocol implementations as suggested by reference [12]), identified as "TCP IW = 2 buf = 64Kbytes". Over the satellite test-bed, the performance is substantially equivalent to IW = 1 for large file transfer (there is an improvement of 2.2% in case of a file of 2.8 Mbytes) but the increase is more relevant for short transfers (14.9 % in case of a 100 Kbytes file) [18].

TABLE 1 : Overall transfer time and average throughput, 2.8 Mbytes, "TCP IW=1 buf=64 Kbytes" as reference.

TCP configuration	Transfer time [s]	Average throughput [Kbytes/s]	Gain[%]
TCP IW = 1 buf = 64 Kbytes	49.21	56.7	–
TCP IW = 6 buf = 320 Kbytes	13.96	199.8	71.6
TCP CONST K = 2	13.22	211.0	73.1
TCP CONST K = 4	12.65	220.4	74.3
TCP LINEAR thr = 10	13.81	201.9	71.9
TCP LINEAR thr = 20	13.16	212.0	73.3
TCP LINEAR thr = 50	12.80	217.8	74.0
TCP MTSI	12.57	221.9	74.5

TABLE 2: Overall transfer time and average throughput, 2.8 Mbytes, "TCP IW=6 buf=320 Kbytes" as reference

TCP configuration	Transfer time [s]	Average throughput [Kbytes/s]	Gain[%]
TCP IW = 6 buf = 320 Kbytes	13.96	199.8	–
TCP CONST K = 2	13.22	211.0	5.3
TCP CONST K = 4	12.65	220.4	9.4
TCP LINEAR thr = 10	13.81	201.9	1.0
TCP LINEAR thr = 20	13.16	212.0	5.7
TCP LINEAR thr = 50	12.80	217.8	8.3
TCP MTSI	12.57	221.9	10.0

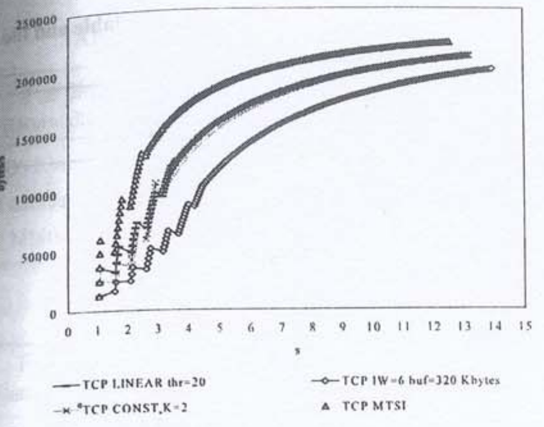

Fig 9 Instantaneous throughput (bytes/s) versus time, "TCP MTSI" compared with other F(·) implementations, single-connection case, 2.8 Mbytes

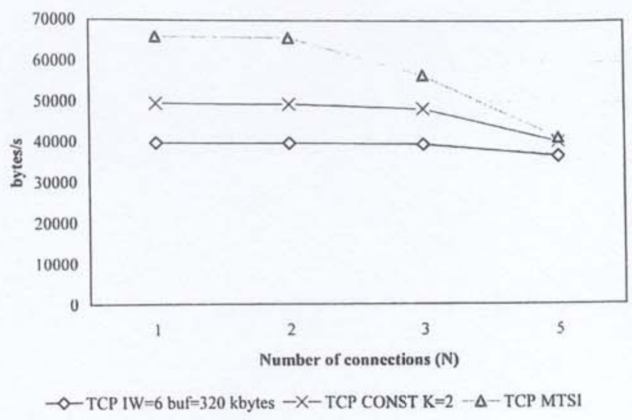

Fig 10 Throughput per connection (bytes/s) versus the number of connections, multi-connection case, 100 Kbytes

- TCP (Initial Window IW = 6 and buffer set to 320 Kbytes), identified as "TCP IW = 6 buf = 320Kbytes". The aim of this configuration is to isolate exactly the real contribution of the new scheme, not considering the advantage already given by the extension of the buffer length and initial window.
- "TCP Westwood+" (by using a buffer length of 64 Kbytes and IW = 3, as set in the available version of this protocol).
- "Ideal" that sets the best possible transfer time by considering an overhead of 52 bytes (out of 1500 bytes) for IP and TCP headers. The overall bandwidth available at the IP layer is decreased of 3.46%. An acknowledgement mechanism is still used. Transfer time evaluated, as said in section 4.1, is time elapsed from the first packet sent to the last acknowledgement received. It is computed as: $\frac{\text{File dimension}}{\text{Effective bandwith}} + \text{RTT}$, where the effective bandwidth is the available bandwidth considering the overheads and RTT is the round trip time, which takes into account the propagation delay for information and acknowledgements. In facts, if a file of 3 Mbytes is transferred over as bandwidth of 2 Mbits/s:

$$\text{Transfer time} = \frac{3145056 \cdot 8\ \text{bits}}{2000000 \cdot 0.965\ 3\ \text{bits/s}} + 0.511\text{s} = 13.54\text{s}.$$

Table 3 contains the overall transmission time (measured in seconds) for the different solutions at the transport layer identified above.

Table 4 contains the comparison of "TCP MTSI" with respect to "TCP IW = 2 buf = 64 Kbytes". The adaptation ("TCP MTSI") of the slow start algorithm, including the increase of the initial window and of the source/destination buffer, guarantees an outstanding performance improvement with respect to ("TCP IW = 2 buf = 64 Kbytes". If the bandwidth is limited, the gain is very relevant only for small files but, as soon as the bandwidth available increases, the gain raises and reaches about 75% for a 2 Mbits/s of bandwidth and a file of 10 Mbytes. For a very small file, the gain is mainly due to the initial window and to the MTSI scheme, while, for big file transfers much is due to the buffer dimension.

The exact role of the new algorithm may be seen by comparing "TCP MTSI" with "TCP IW = 6 buf = 320 Kbytes", which implements exactly the same buffer and initial window of "TCP MTSI" but differs from it for the slow start increasing function. The comparison is shown in Table 5 that contains the gain of "TCP MTSI" by varying bandwidth available and file dimension and having "TCP IW = 6 buf = 320 Kbytes" as reference. For very small file dimension there is no gain because the first package of sent packets contained in the initial window is almost sufficient to conclude the file transfer. As a consequence, MTSI algorithm has no role at all. Concerning the other cases: the gain decreases when the file dimension increases and is, in percentage, less relevant for a file transfer of big dimension because the new algorithm acts more effectively in the first part of the communication where the help of the extended buffer is negligible. The gain offered by "TCP MTSI" for transfers of 100 Kbytes is noticeable.

Table 6 contains the gain of "TCP MTSI" taking "TCP Westwood+" as reference. It is necessary to remind that TCP Westwood tests are performed by using a Linux 2.4 operating system kernel and without any change concerning initial window and buffer. It means that the following parameters are set: IW = 3, buf = 64 Kbytes. The choice is justified by the will of using the original version of "TCP Westwood+" and by the current availability of this software. Differently from kernel 2.2, version 2.4 does not half the receiver window dimension. The overall effect is equivalent to set buf = 128 Kbytes in kernel version 2.2. The comparison has the aim of highlighting the

TABLE 3: Overall transmission time [s] for different transport layer implementations, by varying bandwidth available and file dimension

		TCP IW = 2 buf = 64Kbytes	TCP IW = 6 buf = 320Kbytes	TCP Westwood+	Ideal	TCP MTSI
Bandwidth [bit/s]	*File [bytes]*	*Time [s]*	*Time [s]*	*Time [s]*	*Time [s]*	*Time [s]*
500 K	10 K	1.66	1.09	1.19	0.68	1.09
	100 K	4.28	3.18	3.69	2.22	2.64
	1 M	21.19	18.89	19.4	17.89	18.33
	3 M	58.7	53.71	54.22	52.64	53.16
	10 M	189.96	175.63	176.15	174.32	175.08
1 M	10 K	1.61	1.07	1.11	0.60	1.07
	100 K	3.95	2.86	3.34	1.36	1.83
	1 M	20.22	10.76	11.44	9.20	9.7
	3 M	56.16	28.04	29.38	26.58	27.13
	10 M	181.67	89.25	92.25	87.42	88.3
2 M	10 K	1.58	1.05	1.08	0.55	1.05
	100 K	3.8	2.71	3.23	0.94	1.69
	1 M	19.75	7.27	11.19	4.86	5.73
	3 M	54.91	16	28.75	13.54	14.46
	10 M	177.54	46.55	90.06	43.96	44.99

TABLE 4: Gain ("TCP IW=2 buf=64 Kbytes", as reference) by varying bandwidth available and file dimension

File dimension Bandwidth available	10 Kbytes	100 Kbytes	1 Mbyte	3 Mbytes	10 Mbytes
500 Kbits/s	34.34%	38.32%	13.50%	9.44%	7.83%
1 Mbit/s	33.54%	53.67%	52.03%	51.69%	51.40%
2 Mbits/s	33.54%	55.53%	70.99%	73.67%	74.66%

characteristics of "TCP MTSI". "TCP Westwood+" behaviour, if there are no errors, is exactly the same of TCP New Reno so the gain shown by "TCP MTSI" in Table 6 is due to the effect of MTSI algorithm, together with an extended initial window and buffer, similarly as commented in Table 4.

Table 7 compares "TCP MTSI" with the "Ideal" transport protocol that uses all the bandwidth available considering the TCP and IP header overheads. The gain shown, in this case, is negative and measures the performance decrease of "TCP MTSI" with respect to an ideal protocol. "TCP MTSI" is always very efficient for big file transmissions. It keeps a good quality if the bandwidth is limited but it is far from the ideal performance when a short file is sent through a large bandwidth. It means that, even if the new algorithm gives great advantages with respect to other transport layer schemes, it cannot use all the available bandwidth over the channel. MTSI slow start is always a probing scheme and cannot quickly reach the maximum available bandwidth. The effect on the overall transmission time may be observed in Table 3.

Figure 11 and Fig 12 contain, respectively, the throughput per connection and the overall throughput by varying the number of connections, in presence of multiple connections and 625 Kbytes/s of available bandwidth. "TCP MTSI" is compared with "TCP IW = 2 buf = 64 Kbytes", "TCP IW = 6 buf = 320 Kbytes", "TCP Westwood+" and "Ideal". Even if quite far from the ideal protocol, "TCP MTSI" shows a relevant advantage both for 3 and 5 connections. "TCP MTSI" performance with 10 connections is slightly above "TCP IW = 6 buf = 320 Kbytes" and "TCP Westwood+" performance.

The "distance" between "TCP MTSI" and the ideal protocol shows the improvement margin that a new transport layer, based on acknowledgement scheme, can have.

4.3.2. *Packet loss*

Packet loss is uniformly distributed with average 0.01 (1%). Table 8 contains the overall transmission time for the configurations "TCP IW = 2 buf = 64 Kbytes", "TCP IW = 6 buf = 320 Kbytes", "TCP Westwood+", "Ideal" and "TCP

TABLE 5: Gain ("TCP IW=6 buf=320 Kbytes", as reference) by varying bandwidth available and file dimension

File dimension / Bandwidth available	10 Kbytes	100 Kbytes	1 Mbyte	3 Mbytes	10 Mbytes
500 Kbits/s	0.00%	16.98%	2.96%	1.02%	0.31%
1 Mbit/s	0.00%	36.01%	9.85%	3.25%	1.06%
2 Mbits/s	0.00%	37.64%	21.18%	9.62%	3.35%

TABLE 6: Gain ("Westwood+", as reference) by varying bandwidth available and file dimension

File dimension / Bandwidth available	10 Kbytes	100 Kbytes	1 Mbyte	3 Mbytes	10 Mbytes
500 Kbits/s	8.40%	28.46%	5.52%	1.95%	0.61%
1 Mbit/s	3.60%	45.21%	15.21%	7.66%	4.28%
2 Mbits/s	2.78%	47.68%	48.79%	49.70%	50.04%

TABLE 7: Gain ("Ideal", as reference) by varying bandwidth available and file dimension

File dimension / Bandwidth available	10 Kbytes	100 Kbytes	1 Mbyte	3 Mbytes	10 Mbytes
500 Kbits/s	−60.53%	−19.19%	−2.48%	−0.99%	−0.44%
1 Mbit/s	−79.83%	−34.26%	−5.45%	−2.09%	−1.01%
2 Mbits/s	−89.87%	−80.36%	−18.02%	−6.77%	−2.34%

TABLE 8: Overall transmission time [s] for different transport layer type, varying bandwidth and file dimension, 1% packet loss

Bandwidth [bit/s]	File [bytes]	TCP IW = 2 buf = 64Kbytes Time [s]	TCP IW = 6 buf = 320Kbytes Time [s]	TCP Westwood+ Time [s]	Ideal Time [s]	TCP MTSI Time [s]
500 K	1 M	43.55	34.59	26.55	18.06	18.962
	3 M	125.96	106.15	76.51	53.17	55.713
1 M	1 M	39.51	22.77	19.68	9.29	13.807
	3 M	121.75	103.47	59.79	26.84	39.92
2 M	1 M	41.05	21.42	19.24	4.90	12.648
	3 M	124.98	110.28	54.89	13.67	31.151

MTSI", varying bandwidth and file dimension.

Figure 13 shows the gain of the configurations "TCP IW = 6 buf = 320 Kbytes", "TCP Westwood+", "TCP MTSI" and "Ideal", having "TCP IW = 2 buf=64 Kbytes" as reference and varying bandwidth available and file transfer dimension. The gain of "TCP MTSI" ranges from 56.5% up to 75.1%.The performance of "TCP MTSI" is above "TCP Westwood+" and not so far from the ideal.

Figure 14 reports the gain of "TCP Westwood+", "TCP MTSI" and "Ideal", taking "TCP IW=6 buf=320 Kbytes" as reference. "TCP MTSI" gain, which ranges from 45.18% to 71.75%, is really relevant. The results allow showing the limited role of the initial window and of the buffer length in presence of packet loss: the advantage provided by "TCP MTSI" is outstanding also for big transfers with relatively low bandwidth. Moreover, the advantage of "TCP MTSI" in presence of errors is much more relevant than in the ideal channel case and the performance improvement increases with the dimension of the file transfer.

Figure 15 allows focusing on the gain of "TCP MTSI" and "Ideal", compared with "TCP Westwood+". The gain of "TCP MTSI", more limited than in the previous cases, is always meaningful and ranges from 27.18% to 43.25%.

The improvement margin of the proposed strategy may be seen in Table 9, where the gain of "TCP MTSI" by taking "Ideal" as reference and varying file dimension and bandwidth available, is shown. The performance, even if very satisfying, may be improved for large bandwidth

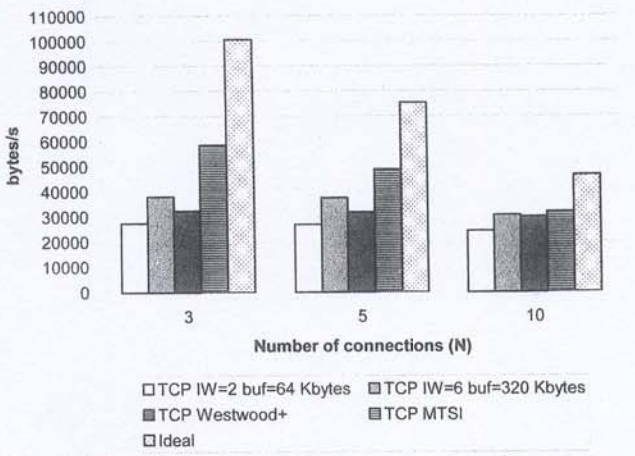

Fig 11 Throughput per connection versus number of connections, multiple connection case, 625 Kbytes/s

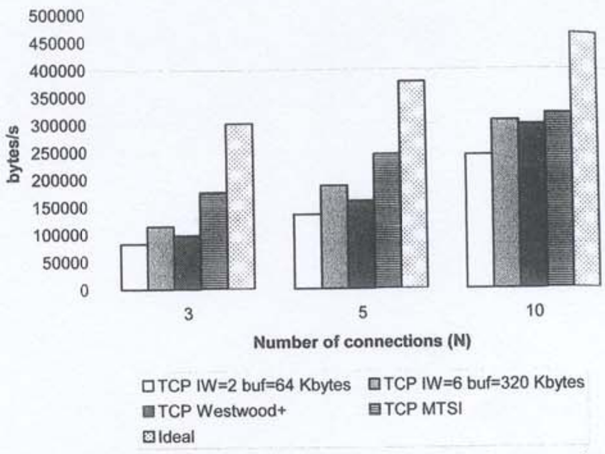

Fig 12 Overall throughput versus number of connections, multiple connection case, 625 Kbytes/s

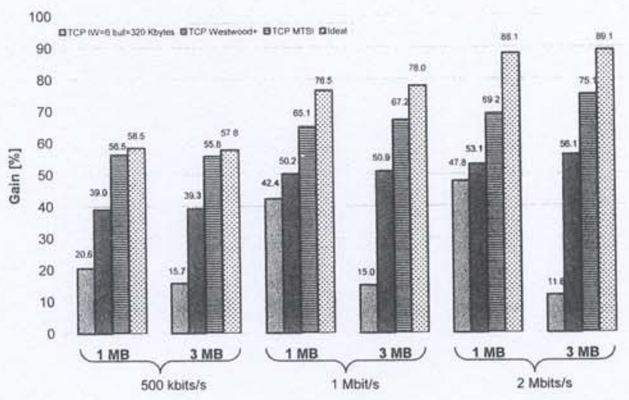

Fig 13 Gain ("TCP1W = 2 buf = 64 Kbytes/s" as reference) vs file dimension and bandwidth – 1% packet loss

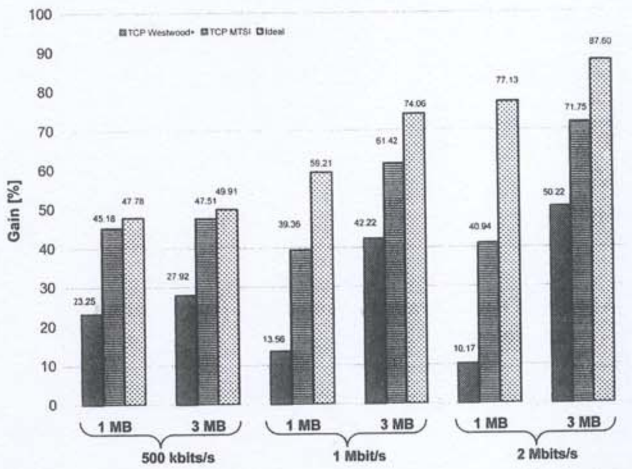

Fig 14 Gain ("TCP1W = 6 buf = 320 Kbytes/s" as reference) vs file dimension and bandwidth – 1% packet loss

availability. The drawbacks is in the congestion avoidance scheme that still considers a loss only due to congestion by following a TCP- like scheme. Actually, over satellite channels, loss is mainly due to transmission errors. In this context, the reduction of the transmission window imposed by the congestion avoidance phase is not useful to reduce errors because, actually, there is no congestion. "TCP MTSI" uses the same mechanism of TCP but, increasing the transmission window much more than TCP at the beginning of the connection, it limits the negative effect of the window reduction because, when the reduction is performed, the window value is higher. This result may be improved by changing the window control in case of loss. The congestion avoidance algorithm should be substituted with another scheme but, in the case, a complete change of the protocol is implied. Further research may investigate alternatives in this direction.

It is interesting also evaluating the robustness against channel errors of the new solution. Table 10 contains the results of the ideal channel case ("No loss"), of the 1% loss case ("Loss") as well as of the performance difference, in percentage, between the two results, for the different configurations. "TCP MTSI" is quite robust due to the new transmission window increasing mechanism. When the bandwidth available is 500 Kbits/s, the difference between the "No loss" and "Loss" case is limited. It increases when the available bandwidth is larger. As said above, the problem is due to the interpretation of the losses.

5. CONCLUSIONS

The paper has introduced an increase function to be used within the TCP slow start algorithm to limit the inefficiencies of the TCP over large delay per bandwidth satellite links. The scheme is called "TCP MTSI".

The tests have been performed by using both a real test-bed and a satellite emulator. File transfer, performed by an ftp-like tool, has been used as reference application. Both the satellite channel available bandwidth and the dimension of the file to be transmitted have been varied during the performance analysis, which has been performed both when only one connection shares the network and when there are more connections in the network.

The behaviour of "TCP MTSI" is compared both with TCP configurations (implementing extended buffer and initial window), which allow to isolate the real contribution of the new scheme, and with alternative solutions. The investigation is carried out both in case of ideal channel and in case of errors (1% packet loss).

The advantages provided by "TCP MTSI" are really relevant both in the ideal and errored channel case. The drawback of "TCP MTSI" is that the algorithms used, even if modified, have the same structure as in TCP. It does not use all the bandwidth available: it increases its sending rate depending on the number of received acknowledgements, considers all the errors as congestion and reduces the information rate by using congestion avoidance. Future studies may try to overcome the mentioned drawbacks by designing a different protocol stack operating in strict connection with a bandwidth allocation scheme.

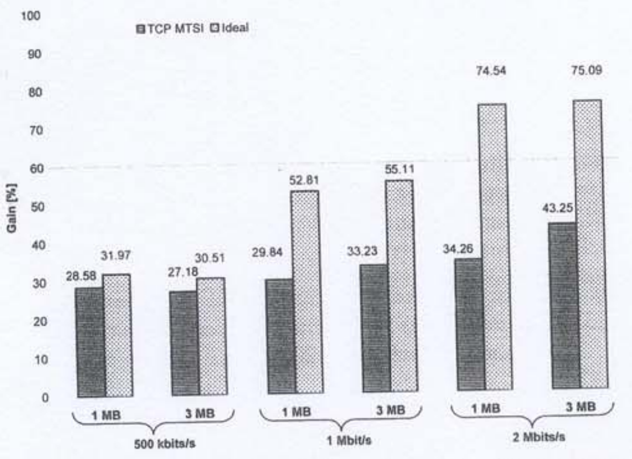

Fig 15 Gain ("TCP Westwood+" as reference) vs file dimension and bandwidth –1% packet loss

TABLE 9: Gain ("Ideal", as reference) by varying file dimension and bandwidth available —1% packet loss

File dimension / Bandwidth available	1 Mbyte	3 Mbytes
500 Kbits/s	– 4.98%	– 4.79%
1 Mbit/s	– 48.67%	– 48.74%
2 Mbits/s	– 158.18%	– 127.80%

ACKNOWLEDGEMENTS

The author would like to thank his colleagues Dr Tomaso de Cola and Dr Giancarlo Portomauro of the CNIT Laboratory "Satellite Multimedia Communications" of the University of Genoa for the work of revision and for their help in the implementation of the emulation tools.

REFERENCES

1. Kota S, Jain R & Goyal R, Broadband Satellite Network Performance, *IEEE Communications Magazine 1999*, vol 37, no 7, pp 94-95.
2. Maral G & Bousquet M. *Satellite Communications Systems - Systems, Techniques and Technology,* IIIrd, John Wiley & Sons: Chichester, England, 1998.
3. Allman M, Glover D & Sanchez L. Enhancing TCP Over Satellite Channels using Standard Mechanism, *IETF, RFC 2488*, January 1999.
4. Partridge C & Shepard TJ, TCP/IP Performance over Satellite Links, *IEEE Network,* vol 11, no 5, pp 44-49, 1977.

TABLE 10 Overall transmission time [s] by varying bandwidth and file dimension, no loss and 1% loss case.

		TCP IW=2 buf=64Kbytes	TCP IW=6 buf=320Kbytes	TCP Westwood+	Ideal	TCP MTSI
Bandwidth [bit/s]	Files [bytes]	Time [s] No loss-Loss Difference%	Time [s] No loss-Loss Difference%	Time [s] No loss-Loss Difference%	Time [s] No loss-Loss Difference%	Time [s] No loss-Loss Difference%
500 K	1 M	21.19-43.55	18.89-34.59	19.40-26.55	17.89-18.06	18.33-18.96
		51.35%	45.39%	26.93%	0.97%	3.33%
	3 M	58.70-125.96	53.71-106.15	54.22-76.51	52.64-53.17	53.16-55.71
		53.40%	49.40%	29.14%	0.99%	4.58%
1 M	1 M	20.22-39.51	10.76-22.77	11.44-19.68	9.20-9.29	9.70-13.81
		48.82 %	52.74%	41.87%	0.94%	29.75%
	3 M	56.16-121.75	28.04-103.47	29.38-59.79	26.58-26.84	27.13-39.92
		53.87%	72.90%	50.86%	0.98%	32.04%
2 M	1 M	19.75-41.05	7.27-21.42	11.19-19.24	4.86-4.90	5.73-12.65
		51.88%	66.05%	41.83%	0.90%	54.70%
	3 M	54.91-124.98	16.0-110.28	28.75-54.89	13.54-13.67	14.46-31.15
		56.06%	85.49%	47.62%	0.96%	53.58%

5. Lakshman TV & Madhow U, The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss, *IEEE/ACM Transactions On Networking*, vol 5, no 3, pp 336-350, 1997.

6. Ghani N & Dixit S, TCP/IP Enhancement for Satellite Networks, *IEEE Communications Magazine*, vol 37, no 7, pp 64-72, 1999.

7. Allman M, Dawkinks S, Glover D, Griner J, Henderson T, Heidemann J, Ostermann S, Scott K, Semke J, Touch J, Tran D. Ongoing TCP Research Related to Satellites, *IETF, RFC* 2760, February 2000.

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

9. Ephremides A, Guest Editor, *International Journal of Satellite Communications* vol 19, no 1, pp 1-139, 2001.

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

11. Jacobson V, Congestion Avoidance and Control, *Proceedings of SIGCOMM '88*, Palo Alto, CA, pp 314-329.

12. Allman M, Paxson V & Stevens WS, TCP Congestion Control, *IETF, RFC 2581*, April 1999.

13. Brakmo LS, O'Malley SW, Peterson LL, TCP Vegas: New techniques for congestion detection and avoidance, *Proceedings of ACM SIGCOMM Symposium*, London, pp 24-35, August 1994.

14. Brakmo LS, Peterson LL & TCP Vegas, End to End congestion avoidance on a global internet, *IEEE Journal an Selected Areas in Communications*, vol 13, no 8, pp 1465-1480, 1995.

15. Floyd S & Henderson T, The NewReno Modification to TCP's Fast Recovery Algorithm, *IETF, RFC* 2582, April 1999.

16. Mathis M, Mahdavi J, Floyd S & Romanow A, TCP Selective Acknowledgment Options, *IETF, RFC* 2018, October 1996.

17. Floyd S, Mahdavi J, Mathis M & Podolsky M. An extension to the Selective Acknowledgment (SACK) option for TCP, *IETF, RFC* 2883, July 2000.

18. Marchese M, TCP Modifications over Satellite Channels, Study and Performance Evaluation. *International Journal of Satellite Communications*, vol 19, no 1, pp 93-110, 2001.

19. Allman M, Floyd S & Partridge C, Increasing TCP's Initial Window, *IETF, RFC* 2414, September 1998.

20. Poduri K & Nichols K, Simulation Studies of Increased Initial TCP Window Size, *IETF, RFC* 2415, September 1998.

21. Allman M, Hayes C & Ostermann S. An Evaluation of TCP with Larger Initial Windows, *ACM Computer Communication Review*, 1998, vol 28, no 3, pp 41-52.

22. Ludwig R & Meyer M, The Eifel Detection Algorithm for TCP, *IETF, RFC*3522, April 2003.

23. Gerla M, Sanadidi MY, Wang R, Zanella A, Casetti C & Mascolo S, TCP Westwood: Congestion window control using bandwidth estimation, *Proceedings of GLOBECOM 2001 - IEEE Global Telecommunications Conference*, San Antonio, TX, pp 1698-1702, November 2001.

24. Ferorelli R, Grieco LA, Mascolo S, Piscitelli G & Camarda P, Live internet measurements using westwood+ TCP congestion control, *Proceedings of GLOBECOM 2002 - IEEE Global Telecommunications Conference*, Taipei, pp 2590-2594, November 2002.

25. Peng Fu C, Liew SC, TCP Veno, TCP Enhancement for Transmission Over Wireless Access Networks, *IEEE Journal on Selected Areas in Communications*, vol 21, no 2, pp 216-228, 2003.

26. Akyildiz IF, Morabito G, Palazzo S. TCP-Peach: A new congestion control scheme for satellite IP networks, *IEEE/ ACM Transactions on Networking*, vol 9, no3, pp 307-321, 2001.

27. Akyildiz IF, Zhang X & Fang J, TCP-peach+, Normal Enhancement of TCP-peach for satellite IP networks, *IEEE Communications Letters*, vol 6, no 7, pp 303-305, 2002.

28. Bharadwaj VG, Baras JS & Butts NP, An Architecture for Internet Service via Broadband Satellite Networks, *International Journal of Satellite Communications*, vol 19, no1, pp 29-50, 2001.

29. Kota S, Pahlavan K & Leppanen P, *Broadband Satellite Communications for Internet Access*, Kluwer Academic Publisher: Norwell, USA, 2004.

30. Border J, Kojo M, Griner J, Montenegro G & Shelby Z, Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations, *IETF, RFC* 3135, June 2001.

31. Henderson TR, Katz RH. Transport Protocols for Internet-Compatible Satellite Networks, *IEEE Journal on Selected Areas in Communications*, vol 17, no 2, pp 326-344, 1999.

32. Consultative Committee for Space Data Systems (CCSDS), Space Communications Protocol Specification-Transport Protocol, *CCSDS* 714.0-B-1, *Blue Book*, May 1999.

33. TC-SES, Satellite Earth Stations and Systems (SES). Broadband Satellite Multimedia, IP over Satellite, *ETSI Technical Report, TR* 101 985, v1.1.2, November 2002.

34. Marchese M, Performance Analysis of the TCP Behavior in a GEO Satellite Environment, *Computer Communications Journal*, vol 24, no 9, pp 877-888, 2001.

35. Marchese M, TCP/IP-*based Protocols over Satellite Systems: A Telecommunication Issue*, Chapter of the book "Reliability, Survivability and Quality of Large Scale Telecommunication Systems", P Stavroulakis, Ed John Wiley & Sons: Chichester, England, pp 167-198, 2003.

36. Davoli F & Marchese M, *SATELLITE SYSTEM SIMULATION, Techniques and Applications*, Chapter of the book, Applied System Simulation, Methodologies and Applications, M Obaidat, Ed Kluwer Academic Publisher, Norwell, pp 155-177, 2003.

AUTHOR

Mario Marchese was born in Genoa, Italy in 1967. He received the Laurea degree (cum laude) from the University of Genoa, Genoa, Italy, in 1992, his qualification as a Professional Engineer in 1992, and a PhD degree in telecommunications from the University of Genoa in 1996. From 1999 to 2004, he worked with the University of Genoa Research Unit, Italian National Consortium of Telecommunications (CNIT), where he was head of research. Since 2005, he has been an Associate Professor in the Department of Communication, Computer and Systems Science (DIST), University of Genoa. He is founder and still technically responsible for the CNIT/DIST Satellite Communication and Networking Laboratory (SCNL), University of Genoa, which contains high-value devices and tools, and implies the management of different units of specialized scientific and technical personnel. He is Vice-Chair of the IEEE Satellite and Space Communications Technical Committee. He is author and co-author of more than 80 scientific papers in international magazines, international conferences and book chapters. His main research interests include satellite networks, transport layer over satellite and wireless networks, quality of service over ATM, IP, and MPLS, and data transport over heterogeneous networks.

* * * *