Pakistan Journal of Information and Technology 1(3): 250-254, 2002

Improving the Performance of TCP in LEO Satellite Environment

[1]A.M. Abdelrahman, M. S. Abdalla, [2]B. M. Ali, [2]V. Prakash and [2]R. K. Z. Sabudin
[1]Faculty of Engineering and Technology, Multimedia University, Malaysia
[2]Computer and Communication System Engineering, University Putra Malaysia

Abstract: Recent growth of the Internet has led the development of services over satellites. Basically, the TCP was proposed as a common transport protocol for the internet over wired networks, but for wireless further modifications are needed. Thus this paper proposes an Enhanced Selective Acknowledgements (ESACK) mechanism to improve the performance of TCP over LEO satellite links. In particular the ESACK detects the congestion by considering losses in two consecutive windows. It assumes that losses in the first window may occur due to transmission errors or handoff, but when losses appear in the second window it considers the possibility of network congestion. The performance of the ESACK has been evaluated extensively over LEO satellites. The results show that the ESACK provides higher congestion window than that of the SACK, which results in improving the effective throughput and reducing the total transmission time. In addition, the ESACK supports fairness among multiple connections.

Key Words: TCP Mechanisms, SACK, LEO Satellites

Introduction

Recent growth of the Internet has led the development of Internet services everywhere and over every possible medium of communications. In this study, LEO satellite has been addressed as an efficient medium for the Internet services to reach remote users. However, the Internet has faced further challenges over satellites, since the standard Internet protocols like Transmission Control Protocol (TCP) have been designed primarily for wired networks. Several problems may arise when the TCP traffic transport over long link satellite networks. The reasoning behind that wireless environments have very different characteristics in terms of latency, jitter, and error rates. Thus for broadband Internet access to continue its fast growth over satellite networks the service providers must overcome the inefficiencies of traditional Internet protocols that degrade the quality of service and inhibit the overall network performance. Despite the progress on improving TCP mechanisms there remain some features that impair the performance over satellite links. The main problem comes from the fact that the TCP operates on the incorrect assumption by relating any loss to the network congestion. Consequently the TCP responses to the loss indication by reducing the congestion window which in turn reduces the performance. While in satellite communications packet errors can occur due to atmospheric and environmental conditions.

This study introduces the standard TCP mechanisms by highlighting their negative impacts on the performance over satellite networks. Then the paper addresses loss indication as the main factor that affects the TCP performance in Low Earth Orbit (LEO) satellite environment. Since losses in LEO may be indicated due to large delay variations, handoffs because of the relative motion of satellites, or transmission errors of satellite links, which cannot be distinguished by the TCP from the network congestion. Therefore an Enhanced SACK (ESACK) mechanism has been proposed to improve the TCP performance over LEO satellite links by considering loss indications in two consecutive windows before assuming that there is network congestion. The ESACK mechanism conceptualises the loss indications in the two windows by assuming that losses in the first window may occur because of non-congestion reasons, but when losses appear in the second window it considers the possibility of network congestion. The proposed ESACK mechanism has been compared with the standard SACK mechanism via extensive simulations. The results obtained show that the ESACK improves the performance over the SACK by enhancing the effective throughput as well as reducing the total transmission time. In addition to that the ESACK supports fairness among multiple connections.

Overview of TCP in Satellite Environment: TCP is the most widely used transport protocol for the Internet applications like File Transfer Protocol (FTP). The TCP protocol divides the transmission message into segments each has a sequence number to identify its order in the data stream. The TCP receiver responds to the successful reception of data by returning an Acknowledgement (ACK) to the sender, which can use this ACK to determine if any data requires retransmission. Thus the TCP was designed and tuned to perform well in fixed networks, where the key functionality is to utilise the available bandwidth and avoid overloading the network (Allman *et al.*, 2000). In this section the major TCP algorithms and their problems in LEO satellite environment are introduced briefly.

- Slow start: the sender window size starts by one segment and increases at a rate of one segment per ACK until either the size of the Congestion Window (CWnd) reaches the Slow Start Threshold (SSThresh) or segment loss occurs. If the segment is lost the sender will assume that the network is congested and immediately reduces its cwnd to one segment and halves the ssthresh (Stevens, 1997).

- Congestion avoidance: Follows the slow start, the cwnd is increased by 1/cwnd for each incoming ACK. The cwnd decreases by a factor of two (cwnd/2) with every loss (Stevens, 1997). Here the problem may occur in LEO satellite links due to unnecessary reduction of the cwnd when encountering non-congestion losses (Cruickshank *et al.*, 2000).
- Fast retransmit: The sender waits for 3 duplicate ACKs and implements fast retransmit. A more precise time taken by the duplicate ACKs to be generated at the receiver is the main problem here. In LEO satellite packet loss may occur due to non-congestion, but the TCP receiver considers any loss due to network congestion and starts sending of duplicate ACKs. This will cause aggressive reduction in the cwnd and the TCP throughput (Floyd, 1996).
- Fast recovery: After a lost segment is resent using fast retransmit the fast recovery is used as follows: the ssthresh is set to half of the current cwnd. The cwnd is set to SSThresh plus 3*segment size. This inflates the cwnd by the number of duplicate ACKs, because each duplicate ACK represents 1 segment that has left the network (Allman *et al.*, 1999). When the next ACK arrives that acknowledges new data, the cwnd is set to the ssthresh and the congestion avoidance is used. Similarly the throughput may degrade due to the decreasing of cwnd (Floyd, 1996).
- Selective Acknowledgements (SACK): When multiple segments are dropped from one window of data, the previous versions of TCP retransmit at most one dropped segment per RTT, as presented by Fall and Floyd (1996). TCP with SACK mechanism is designed to alleviate this problem. The SACK behaves as follows: the receiver waits for the receipt of data to fill the gaps in sequence the space between the received blocks, when missing segments are received it acknowledges them. The sender will have a queue that contains the segments that are transmitted but not yet acknowledged, when receiving an ACK containing a SACK option it will trace which segments must be transmitted. It has been claimed that the best way to improve the fast retransmit and fast recovery algorithms is to use a SACK based algorithm for loss recovery. SACK allows for multiple losses in a transmission window to be recovered in one RTT. However, it was demonstrated by Mathis *et al.*, (1996) that the SACK preserves the congestion control issues present in the standard TCP implementations such as reducing the cwnd in response to any congestion.

Recent advances in satellite technologies together with the fast growth of the Internet have motivated the development of broadband LEO satellite networks such as iridium and teledesic systems which perform onboard switching and signal processing, and being interconnected by Inter-Satellite Links (ISL), the basic architecture was provided by Sturza (1995). Despite the progress on this field the effects of satellite link characteristics on the TCP performance are still remain. Grosser (1999) and Cruickshank *et al.*, (2000) reported some characteristics of LEO satellites: Latency; due to the propagation delay the TCP sender may take long time to determine whether or not a segment has been successfully received at the destination (Gurtov, 2001). Delay variations; because of the relative motion of the LEO satellites the TCP RTT measurements will be affected which may cause false timeouts leading to unnecessary retransmissions (Michael,2000). Handoffs; in LEO systems, inter-plane ISL handoff occurs between different planes at highest latitudes, while cross-seam ISL handoff occurs as satellites pass beyond local horizons. For TCP connection such handoff behaviour may lead to subsequent failure of the data transfer (Huston, 2000). Link outages; as a result of satellite movement during the transmission the TCP data may be lost because there is no link available at a certain time (Huston, 2000). Transmission errors; satellite channels exhibit higher Bit Error Rates (BER) than typical terrestrial networks, but TCP operates on the incorrect assumption that all losses are related to congestion (Huston, 2000).

Several studies reported that TCP over LEO satellite constellation networks have significantly degraded the throughput, the reliability and the efficiency of the TCP protocol (Grosser, 1999). The use of Explicit Congestion Notification (ECN) bit has been studied by Ahmed and Salim (2000), where the sender can receive explicit messages from the routers when congestion occurs. One of the advantages of ECN is that, a TCP connection with a small congestion window can avoid a retransmit timeout. However, Wood (2001a) argued that the ECN cannot completely eliminate packet losses as indications of congestion, and therefore would not allow the end nodes to interpret packet losses as indications of corruption instead of congestion. The TCP sender that has halved its cwnd as a result of packet losses could receive information from the link-level that this packet was lost due to corruption rather than congestion. Thus most of the studies have focused only on congestion issues and still the corruption needs to be addressed (Wood, 2001b).

Recently, Dawkins *et al.*, (2000) have proved that for wired links packet losses due to packet corruption instead of congestion are infrequent; this is not always the case for wireless links. Although many wireless links use Forward Error Correction (FEC) and link-level retransmission to repair packet corruption it is not always possible to eliminate all packet corruption in a timely fashion (Wood, 2001b). Based on this argument the ESACK has been proposed by assuming that there is a high probability of losses due to packet corruption and link outage. In particular it tries to distinguish between network congestion and packet corruption by applying appropriate actions after detecting the losses in two consecutive windows.

Description of Enhanced Selective Acknowledgements: This section introduces the basic concept of the proposed Enhanced Selective Acknowledgements (ESACK) mechanism. The ESACK tries to alleviate the previous problems by enhancing TCP's loss recovery using SACK-based adaptation over LEO satellite links. It tracks losses in every two consecutive windows and assumes that losses in the first window are most probably due to corruption rather than congestion. But when such losses appear in the second window it considers the possibility of network congestion. In a sense that if there is network congestion the probability of losses in more than one window successively is very high.

The ESACK defines the following parameters: Retransmission Time Out (RTO), which is monitored and updated based on Round Trip Time (RTT). Congestion Window (CWnd) specifies the amount of data that the sender can transmit into the network before receiving an acknowledgement. Slow Start Threshold (SSThresh) limits of the slow start phase.

Duplicate Acknowledgements (dupACK) threshold is used to inform the sender that a segment was received out of order as well as indicating what sequence number is expected. Second Window (SecWnd) flag is proposed in this study to keep the track of window with losses at the sender. Initially the secwnd is set by the sender to 1, when losses occur in the first window it is reset to 0, again it is set to 1 when losses occur in the second window and so on. Thus the secwnd is reset

and set by losses in odd and even windows, respectively. When losses occur in the first window the actions will be taken as follows:

A1. If the dupthresh is passed (dupACK >= dupthresh), while the RTO does not expire and the cwnd is still less than the ssthresh (cwnd < ssthresh), then the TCP sender must implement SACK mechanism to retransmit all lost segments whether one or multiple losses per window. After retransmission the sender continues with the same cwnd size while maintaining the increment rate of one segment per RTT (1 segment / window). The goal behind using the same current window size is to avoid throughput degradation by further reducing the window size, since the connection is still at the slow start where more bandwidth is wasted.

A2. If both dupACK and cwnd exceed their thresholds (dupACK >= dupthresh AND cwnd >= ssthresh), while the RTO does not expire, the sender must implement SACK mechanism to retransmit all lost segments. After retransmission the sender reduces the current cwnd to 60% rather than 50% of its value, while continues in increasing the cwnd at a rate of one segment per window (1 segment / window). The reduction to 60% of the cwnd size is approximated through a series of simulations by using the original SACK for different file sizes at high link losses. These simulations result in an average throughput of 60% from the available bandwidth. Hence, the enhancement assumes that at least 60% of the current throughput should be maintained.

A3. If the RTO expires, regardless the status of the other parameters the sender must ignore prior SACK information in determining which data to retransmit and retransmit all segments at the left edge of the window as described in. This will trigger the use of the basic slow start mechanism where the value of the ssthresh is halved and the cwnd is set to one segment. Normally the RTO is set to a relatively higher value which may rarely expire.

The actions in the next successive window will be taken, if one of the above cases is repeated, such as follows:

B1. If the same conditions are observed as in A1 the same actions will be taken, but by setting the cwnd to 50% of its current value.

B2. If the same conditions are repeated as in A2 similar actions will be taken, but by setting the cwnd to 50% of the original value of the ssthresh.

B3. If the RTO expires, similar actions will be taken as in A3.

These groups of actions will be considered throughout the transmission time and will be repeated alternatively In all windows with losses. Thus the actions A1, A2 and A3 will be carried when losses occur in odd windows. Similarly, the actions B1, B2, and B3 are implemented when losses occur in even windows.

Results and Discussion

This section presents and discusses the results obtained from a number of simulation scenarios using SACK and ESACK. The Network Simulator (NS) has been selected for all simulation scenarios, while the Teledesic satellite system (288 satellites) is used to link a source in Berkley (37.9 latitude, -122.3 longitude) with a destination in Kuala Lumpur (3.7 latitude, 100.19 longitude). The link is full duplex that is 1.5 Mbps up/down, 155Mbps inter-satellite, and a drop tail queuing with a queue limit of 50 packets. The link is modelled with the Bit Error Rate (BER) ranges from 10^{-9} to 10^{-6}. The following FTP file sizes have been transferred: 5, 10, and 20 Mbytes, at a packet length of 512 bytes. The ssthresh is set to 70 segments, as a maximum cwnd size supported by 1.5 Mbps. Also the ssthresh is decreased to 18 segments for multiple connections from different sources in South Africa contending for the same destination in Kuala Lumpur.

Results of Different File Sizes: Fig. 1 shows the cwnd for FTP file sizes 5 and 20MB at a BER of 10^{-7}. Initially all graphs experience slow start mechanism by increasing the cwnd exponentially. Similarly they show the same cwnd values for both SACK and ESACK during the congestion avoidance phase before the occurrence of the first loss. However, after the first loss each graph behaves differently for SACK and ESACK. Since the SACK responses to any packet loss by halving the current cwnd if it is less than the Slow Start Threshold (SSThresh), otherwise it sets the cwnd to half of the ssthresh. On the other hand, ESACK treats the packet losses differently by dealing with losses in any two successive windows throughout the transmission. At the first loss it sets the cwnd to 60% of its current value only if it is greater than the ssthresh, otherwise it uses the current cwnd value as it is. At the second loss it sets the cwnd in the same way as the SACK does. It can be seen that the first loss occurs at ≈ 5sec where the cwnd is greater than the ssthresh (70 segments). Thus the SACK reduces its cwnd to 35, while the ESACK reduces the cwnd to ≈ 49. The second loss occurs at ≈ 6sec where each of the mechanisms halves its current cwnd since both of them are below the ssthresh. This produces two cwnd values: 20 and 26 for SACK and ESACK, respectively. Similarly SACK has done the same by halving the current cwnd at the third and fourth losses. ESACK does not reduce the current cwnd at the third loss since it is less than the ssthresh, but it halves the current cwnd at the fourth loss. It is observed that the overall trend of the cwnd of the ESACK is higher than that of the SACK, which in turn improves the effective throughput as well as reduces the total transmission time at the expense of slightly increases end to end packet delays. Fig. 2 proves that the ESACK gives higher throughput than the SACK one. Also it is shown that the throughput increases with the increase of the file size for both ESACK and SACK. However, the throughput differences between ESACK and SACK almost show similar trends for all file sizes.

Results of Various Bit Error Rates: This subsection examines the data flow for an FTP file size of 10MB under BER of 10^{-7} and 10^{-9} which are represented as high BER and low BER, respectively. Fig. 3 shows slight variations in cwnd between ESACK and SACK mechanisms at low BER. It has been observed that these cwnd variations increase as the number of losses increases, i. e. as the BER increases. Again this dynamic behaviour of the cwnd produces similar variations between ESACK and SACK mechanisms in terms of effective throughputs. Fig. 4 indicates that the overall effective through decreases as the BER increases, but at the same it reflects the behaviour of ESACK against losses, i.e. ESACK provides more throughput over SACK as the number of losses increases. Consequently the overall transmission time increases by the increment of losses, while the ESACK minimises the transmission time more effectively compared to the SACK.

Results of Multiple Connections: The congestion control behaviour has been tested with multiple connections for both mechanisms. Four TCP connections have been established to transfer FTP files, each of size 5MB, from four sources located at different places in South Africa to their destination in Kuala Lumpur. All connections share the same downlink and

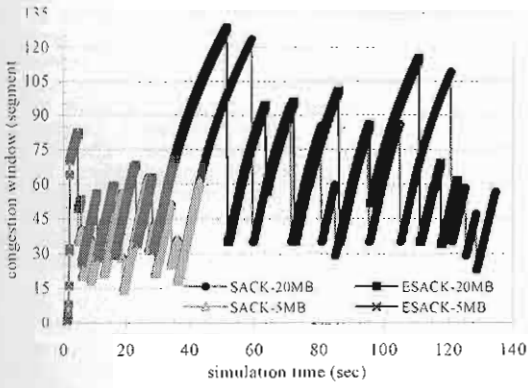

Fig. 1: Dynamics of Congestion Windows for Different File izes(BER≈10^{-7})

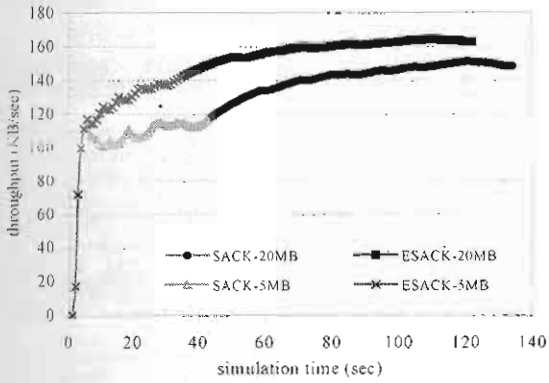

Fig 2: Effective Throughputs for Different File Sizes (BER≈10^{-7})

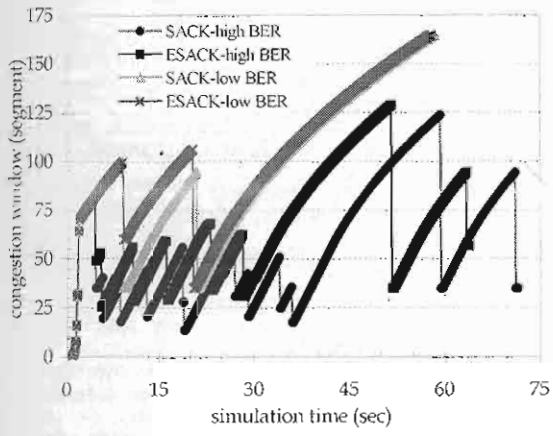

Fig. 3: Dynamics of Congestion Windows for File Size of 10MB (high BER≈10^{-7}, low BER≈ 10^{-9})

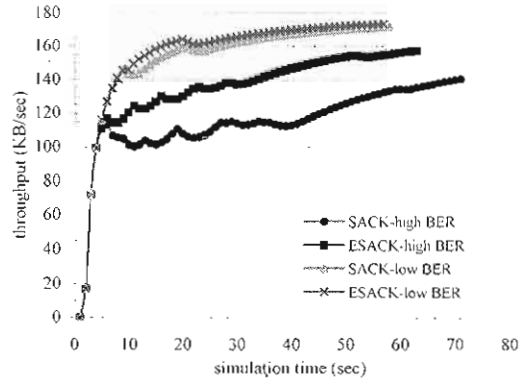

Fig. 4: Effective Throughputs for File Size of 10MB (high BER≈10^{-7}, low BER≈ 10^{-9})

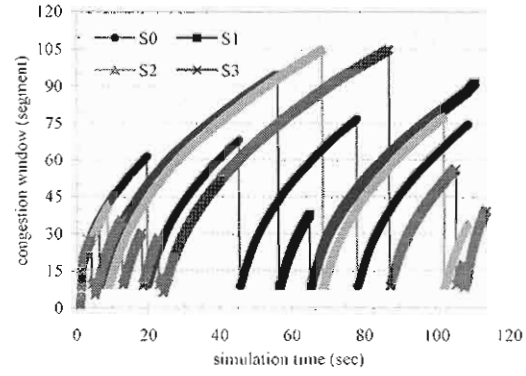

Fig. 5: Dynamics of Congestion Windows for Multiple Sources Using SACK (BER≈10^{-7})

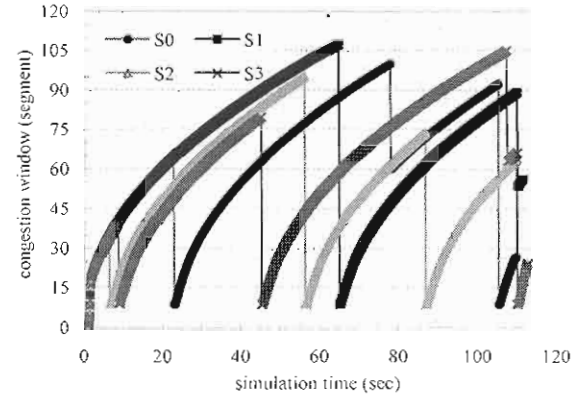

Fig 6: Dynamics of Congestion Windows for Multiple Sources Using ESACK (BER≈10^{-7})

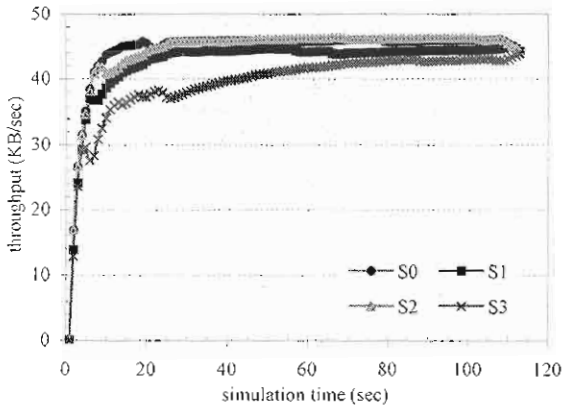

Fig.7: Effective Throughputs for Multiple Sources Using SACK(BER≈10^{-7})

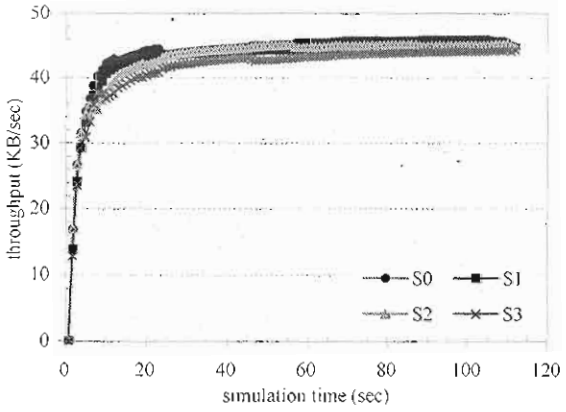

Fig 8: Effective Throughputs for Multiple Sources Using ESACK(BER≈10^{-7})

contend for the same destination. The dynamics of cwnd have been represented in Fig. 5 and 6 for SACK and ESACK, respectively. It can be seen that each of the mechanisms gives similar congestion windows to each of the four connections. Fig. 7 and 8 show the effective throughputs for the four connections provided by SACK and ESACK, respectively. The SACK provides slightly different effective throughputs for the connections, as seen in Fig.7, where the standard deviation is 2.4. However, Fig. 8 shows that the overall gains of connections obtained by the ESACK are quite close, with the standard deviation of 1.1. This indicates that ESACK supports fairness among multiple connections.

Conclusion

This paper investigates the negative impacts of the TCP mechanisms over LEO satellite links. The TCP mechanisms assume that any loss indication is due to the network congestion, and accordingly they reduce the congestion windows which in turn reduce the effective throughputs. This affects the performance more drastically in LEO satellite environments where losses may occur due to delay variations, handoffs, or transmission errors.

In particular this paper addresses losses in LEO satellites as the main problem of TCP and hence proposes the Enhanced Selective Acknowledgements (ESACK) mechanism that improves the performance of TCP over LEO satellite links. The ESACK considers losses in two consecutive windows. It assumes that the losses at the first window may occur due to non-congestion reasons. But when the losses occur in the second window it considers the possibility of network congestion by taking more protective actions. The proposed mechanism has been evaluated by transferring FTP data files, using different sizes and at various bit error rates. When the ESACK mechanism compared with the standard SACK mechanism the following remarks are observed: ESACK provides higher dynamics of congestion windows. The ESACK has shown better average effective throughputs. The transmission times are decreased by ESACK at the expense of slightly increases end to end packet delays. Also the ESACK supports the fairness among multiple connections more effectively.

It can be claimed that with possible slight changes to the SACK the proposed ESACK mechanism improves the overall performance of TCP over LEO satellite links. It protects the TCP throughput over LEO satellites by saving the connection from aggressive reductions of congestion windows when there are non-congestion losses, at the same time it takes appropriate actions when there is a high probability of network congestion.

References

Ahmed, U. and J. Salim; 2000. Performance Evaluationof Explicit Congestion Notification (ECN) in IP Networks. Internet RFC 2884.

Allman, M., D. Glover and L. Sanchez; 1999. Enhancing TCP Over Satellite Channels Using Standard Mechanisms. Internet RFC 2488.

Allman, M., S. Dawkins, D. Glover, J. Griner, D. Tran and T. Henderson, 2000. Ongoing TCP Research Related to Satellites. Internet RFC 2760.

Cruickshank, H., Z . Sun, L. Wood, T. Ors, R. Dhaou, M. Becker and C. Enakis; 1999. Report on LEO Satellite Network Characteristics. BISANTE Consortium.

Dawkins, S., G. Montenegro, M. Kojo, V. Magret and N. Vaidya; 2000. End-to-end Performance Implications of Links with Errors. Internet-draft.

Fall, K. and S. Floyd; 1996. Simulation-based Comparisons of Tahoe, Reno and SACK TCP. Computer Communication Review.

Floyd, S., 1996. Issues of TCP with SACK. Technical report. http://www.aciri.org/floyd/sacks.html

Grosser, P., 1999. Research Proposal. University of South Australia.

Gurtov, A., 2001. Effect of Delays on TCP Performance. Cellular Systems Dev., Sonera, Corporation.

Huston, G., 2000. The Future for TCP. The Internet Protocol Journal, 3 (3).

Mathis, M., J. Mahdavi, S. Floyd and A. Romanow; 1996. TCP Selective Acknowledgment Options. Internet RFC 2018.

Michael, K., 2000. Error Modeling for TCP Performance Evolution. M.Sc thesis, State Univ. of New York,Stony Brook. http://www.cs.sunysb.edu/~kostas/art/MSc/MScThesis.pdf

Stevens, W. R., 1997. TCP Slow Start, Congestion Avoidance, Fast retransmit, and Fast Recovery Algorithms. Internet RFC 2001.

Sturza, M. A., 1995. Architecture of the Teledesic satellite system. Proceedings of the International Mobile Satellite Conference: 212-218.

Wood, L., 2001B. Internetworking with Satellite Constellations. PhD thesis, Centre for Communication Systems Research, School of Electronics, Computing and Mathematics, Univ. of Surrey, Guildford, UK.

Wood, S., 2001A. A Report on Some Recent Dev. in TCP Congestion Control. IEEE Communications Magazine.