An Enhanced TCP Congestion Avoidance Scheme and its Performance Evaluation in High Speed Satellite Networks

Xiaohu Ge*
*Dept. of Electronics and Information Engineering
Huazhong University of Science and Technology
Wuhan, China
*xhge@mail.hust.edu.cn

Frank Y. Li**, and Frank Reichert**
**Dept. of Information and Communication Technology
University of Agder (UiA)
N-4898 Grimstad, Norway
**{frank.li, frank.reichert}@uia.no

Abstract—**High speed satellite communication networks are emerging as part of the future global wireless communication systems. However, existing transmission control protocols for satellite networks do not provide satisfactory performance over high speed satellite links due to their inefficient congestion avoidance algorithms. This paper identifies the reason for low throughput of a widely used protocol Space Communications Protocol Specification (SCPS) in such networks and proposes a new Transmission Control Protocol (TCP) congestion avoidance algorithm to overcome the drawback of the congestion avoidance algorithm used in the SCPS protocol. Numerical results through simulations demonstrate that the proposed new algorithm can achieve significant throughput improvement over links with variable error rates, compared with its legacy counterpart.**

Keywords: Congextion Avoidance Scheme, Satellite Network, TCP, Performance Analysis

I. INTRODUCTION

With the progress in satellite communication technologies, the transmission capacity for satellite networks is steadily increasing, ranging from in the order of hundreds of Mbps up to 1 Gbps. High speed satellite communication networks are sprawling from connecting mainly research organizations to undertaking backbone capacity to cover the globe for commercial purposes in the future [1]. Therefore it is of importance to improve the performance of the traditional TCP to meet the requirements in high speed satellite networks.

As we know, traditional TCPs are developed for terrestrial wire-line networks. Satellite communication links, however, are characteristics of features like long delay and high Bit Error Rate (BER). For example, if the legacy TCP is directly used for satellite networks, the throughput that can be achieved is merely about 200 Kbps, even though the satellite link capacity reaches 1.5 Mbps with a BER at 10^{-6} [2]. The reason for this disappointing result is that the congestion control mechanism in TCP invokes unnecessary rate control, leading to low bandwidth utilization.

To improve TCP throughput performance caused by long delay, many TCP enhanced protocols have been developed. According to different message feedback mechanisms, these enhanced TCP protocols can be categorized into implicit feedback protocols, such as Scalable TCP (STCP) [3], FAST AQM Scalable TCP (FAST TCP) [4], and explicit feedback protocols, such as eXplicit Control Protocol (XCP) [5], and Variable-structure congestion Control Protocol (VCP) [6]. The Westwood protocol is another TCP enhanced protocol based on bandwidth estimation information [7]. On the other hand, protocols considering high BERs have also been proposed. For instance, the Cross-layer Selective ACKnowledgement (CSACK) protocol [8] can improve the throughput performance in satellite networks with high BER, but this proxy-based solution may bring a few robustness and security problems. The TCP-Peach protocol [9] introduced two algorithms, namely sudden start and rapid recovery, in addition to the traditional ones like congestion avoidance and fast retransmission. Simulation experiments show that TCP-Peach can improve satellite network throughput under high BERs [9]. To modify the sudden start and rapid recovery algorithms in traditional TCPs, the TCP-Swift protocol [10] developed two new algorithms, namely speedy start and speedy recovery. In summary, within satellite communication networks, the most successful working protocol so far is SCPS which is mainly developed by NASA and the U.S. DoD. The SCPS suite indeed includes four protocols, and one of these four protocols is the SCPS-Transport Protocol (SCPS-TP) which has been used for implementing TCP in satellite networks. Real-life operational networks demonstrated that the SCPS-TP did improve the throughput for satellite networks used in 1990s [11].

However, the above TCP variants and their enhanced versions do not support high speed satellite communication networks. Even though SCPS-TP can achieve approximately 6.3 Mbps throughput over a 100 Mbps satellite link at the BER of 10^{-8}, the throughput goes down significantly to only 270 Kbps when BER deteriorates to 10^{-6} [12]. Therefore, new TCP schemes and algorithms which can support high speed satellite communication networks need to be developed.

This work is supported by the National Natural Science Foundation of China (No. 60872007) and National 863 High Technology Program of China, (No. 2009AA01Z239).

978-1-4244-4657-5/09/$25.00 ©2009 IEEE ICICS 2009

Considering the fact that a large portion of the throughput is achieved in the SCPS-TP congestion avoidance phase, the efficiency of the SCPS-TP congestion avoidance algorithm is the most critical element for improving throughput in high speed satellite networks. In this paper, we first analyze the problem of the congestion avoidance algorithm in SCPS-TP which limits throughput performance over high speed satellite links, and then develop a new TCP congestion avoidance algorithm to improve the TCP throughput in such networks.

The rest of the paper is organized as follows. Sec. II elaborates the throughput problem in SCPS-TP. Sec. III proposes a new congestion avoidance algorithm to solve this problem. Simulation experiments are carried out in Sec. IV to evaluate the performance of the proposed algorithm. Finally, the conclusions are drawn in Sec. V.

II. PRIMARY PROBLEM ANALYSIS

The most widely used TCP currently deployed in real-life satellite communication networks, SCPS-TP, is modified from legacy TCP. Like a regular TCP process, the SCPS-TP process includes also three phases, namely, slow start, congestion avoidance, and congestion control, of which the congestion control phase includes a fast retransmit algorithm and a fast recovery algorithm. In the congestion avoidance phase of SCPS-TP, a congestion avoidance algorithm has been developed to conservatively control traffic once the *congestion window (cwnd)* exceeds the threshold of the slow start phase [11].

To describe the congestion avoidance algorithm in SCPS-TP, a few basic terms are defined here first: *cwnd* represents the size of the congestion window, Average Round Trip Time (*AvgRTT)* is the average value of the round trip time until currently transmitted segments, and *BaseRTT* is the minimum RTT of the all measured round trip times.

Assume that the network connection is not over-flown. The maximum throughput is given by Maximum_throughput = cwnd/BaseRTT, and the actual throughput is given by Actual_throughput = cwnd/AvgRTT. Let a variable σ be the difference between the maximum throughput and the actual throughput. It can be expressed as σ = Maximum_throughput – Actual_throughput.

Consequently, the congestion avoidance algorithm of the SCPS-TP adjusts the congestion window based on variable σ. In real-life satellite communication networks, two congestion avoidance thresholds, known as α and β, where $\alpha < \beta$, have been set according to the satellite link delay. Let the variable *cwnd_old* be the size of the congestion window before adjustment, and variable *cwnd_new* be the size of the congestion window after adjustment. The congestion window adjusting scheme could be expressed as follows,

$$cwnd\_new = \begin{cases} cwnd\_old + 1, & \text{when } \sigma < \alpha; \\ cwnd\_old, & \text{when } \alpha \le \sigma \le \beta; \\ cwnd\_old - 1, & \text{when } \sigma > \beta. \end{cases} \quad (1)$$

Based on the above expression, it can be observed that the adjustable size of the congestion windows is pretty small in this congestion avoidance algorithm since the value of *cwnd* increases (or decreases) only *additively* as α passes the thresholds. This algorithm is helpful to avoid congestion in the network and the size adjusting rate matches the satellite communication speed in 1990s.

However, when the satellite communication capacity is increased to 100 Mbps, simulation experiments showed that the SCPS-TP network throughput can only increase to approximately 6 Mbps at a very good channel condition, with the BER at 10^{-8} [12]. This result illustrates that SCPS-TP takes too much time to increase its throughput until the maximum capacity is utilized in high speed links. Through simulation experiments to be described in Sec. IV, we observe that the maximum reliable transmission data in one RTT will further increase with the satellite transmission rate when techniques such as window scaling, header compression, Selective Negative ACKnowledgement (SNACK) and Explicit Congestion Notification (ECN) are activated in SCPS-TP. Even if it is possible to adjust congestion avoidance thresholds α and β, SCPS-TP will still increase its congestion windows slowly by one at every RTT according to Eq. (1). As a consequence, the benefit of high speed link is not utilized.

For example, in order for SCPS-TP to increase its congestion windows to reach full utilization of 1 Gbps capacity with packet length of 1500-bytes, it requires over 8.333 RTTs without packet loss. This equals to approximately 45 minutes considering Geosynchronous Orbit (GEO) satellite networks with typically 500 ms propagation delay. For this reason, the throughput in the congestion avoidance phase has been suppressed to a low level even through the link capacity achieves 100 Mbps. We refer to this phenomenon as the bottleneck effect of SCPS-TP in high speed satellite communication networks. This bottleneck explains why the SCPS-TP throughput is far from satisfactory in such networks.

According to the above problem analysis, we believe that the congestion avoidance algorithm in SCPS-TP needs be re-designed to improve its throughput. Especially, the congestion window adjusting scheme must be modified in order to match the high speed links.

III. THE PROPOSED ALGORITHM

In this section, we modify the congestion window adjusting scheme in the congestion avoidance algorithm of the SCPS-TP and propose a new TCP congestion avoidance algorithm based on the measured dynamic network transfer delay and the BER of the link. The network transfer delay is the sum of propagation delay, transmission delay and queuing delay in all network nodes between source and destination. The propagation delay is considered as constant for each type of satellite, but the total transfer delay is variable due to difference transmission delays and queueing delays. Therefore, the main task of our algorithm is to design an efficient congestion algorithm by taking into account the variable transfer delay of the transmitted packet.

In the proposed TCP congestion avoidance algorithm, the congestion avoidance phase has been divided into three sub-phases. In every sub-phase the congestion windows have been adjusted by different functions. Before describing the congestion window adjusting scheme, let us introduce a few variables as follows.

Unit_TD is the half of the value for the minimum RTT among all the RTTs measured during the previous observation period; *cwnd_lim* is the maximum size of the congestion windows and is set as a constant, decided by the packet format; *increment* is the adjusting size of congestion windows; *p* is the BER of the satellite link, which is a variable. A sliding window is used for observation so that these measured values are recorded and updated for ten observation periods. Considering the difference between *AvgRTT* and *Unit_TD* in each observation period*, BaseTD* is defined as the minimum value of all measured delay differences during the past ten observation periods, and it is given by:

$$BaseTD = \min\{AvgRTT - Unit\_TD\}\,, \qquad (2)$$

where TD is the current transfer delay which is given by:

$$TD = AvgRTT - Unit\_TD\,. \qquad (3)$$

Considering the relationship between *BaseTD* and *TD*, the congestion window adjusting scheme has been designed as follows,

$$increment = \begin{cases} \min(S, cwnd\_\lim - cwnd), & TD = BaseTD \\ S * \left(1 - \dfrac{TD - BaseTD}{BaseTD}\right), & BaseTD < TD < 2 * BaseTD \\ 0, & TD \geq 2 * BaseTD \end{cases} \qquad (4)$$

where *S* is a scaler, used for adjusting increment rate of the congestion window. It indicates that our algorithm will adjust the window size *multiplicatively*, while the legacy algorithm in Eq. (1) gives only *additional* adjustment of the window size. Empirically in our design, *S* is obtained by the following equation:

$$S = -A * \lg p\,, \qquad (5)$$

where *A* a pre-determined and optimally-adjusted parameter, obtained through extensive simulations [1].

The reason that we propose *S* as a logarithmic function of *p* is due to the consideration that the window size should be adjusted much faster as compared with Eq. (1) and this adjustment should be related to the link condition. On the other hand, it is not reasonable to set *S* as for instance a linear function of the link condition which is typically represented by BER. Therefore, using a logarithm function suits the purpose quite well. Considering that the nominal BER of a contemporary satellite link is in the order of 10^{-8}–10^{-5}, a combination of a scaler and the logarithmic function leads to Eq. (5) in our proposed scheme.

We are aware of that in many traditional TCP protocols the congestion avoidance algorithm is adjusted based on Packet Loss Ratio (PLR). However, in the satellite network protocols, recovery schemes in the data link layer, such as Automatic Repeat reQuest (ARQ) and Forward Error Correction (FEC) schemes may lead to different complex relation models between BER and PLR. To simply the expression without missing the relationship between *S* and link condition, we directly build the relationship between the window adjusting rate of the congestion windows and BER as expressed in Eq. (5). In real-life applications, TCP layer's algorithms could obtain the PLR information by cross-layer mechanisms, such as discussed in [13].

Furthermore in our algorithm, the congestion window adjusting scheme has been divided into three sub-phases, as shown in Eq. (4). More specifically, in the case of $TD = BaseTD$, it indicates that the network link is in an ideal phase. We call this phase as the no-congestion sub-phase. In this sub-phase, the increment of the congestion windows could be set as *S* if the difference of *cwnd_lim* and *cwnd* is larger than *S*. In the case of $BaseTD < TD < 2 * BaseTD$, it indicates that the network link is in the potential congestion phase. We call this phase as the pre-congestion sub-phase. In the pre-congestion sub-phase, the increment of congestion windows proportionally decreases with the increase of *TD*, as illustrated in Fig. 1. In the case of $TD \geq 2 * BaseTD$, it implies that the network transfer delay is pretty large. Although the network congestion has not happened yet, it has high probability that the network congestion level will arise. In this case, the increment of congestion windows should equate to zero to avoid network congestion. Furthermore when congestion happens, the window size is reduced according to the standard TCP algorithm for window size adjustment.

The proposed congestion window adjusting scheme allows bandwidth probing to be more effective when the current window size gets closer to the ideal point or the BER is low, and become less aggressive as the current window size gets closer to the saturation point or the BER is high.

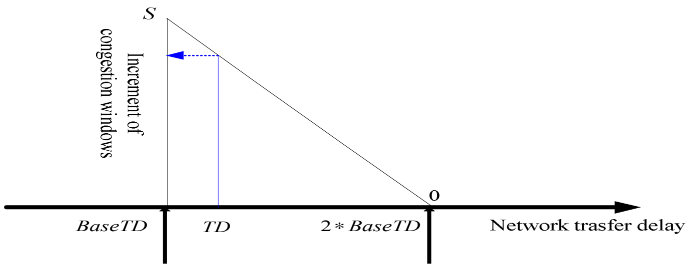

Fig. 1. Increment of the congestion windows in the pre-congestion sub-phase.

Based on the congestion windows adjusting scheme in the congestion avoidance phase, the proposed congestion avoidance algorithm is listed below as Algorithm 1.

---

[1] The value of *A* does have effect on the performance of the algorithm. Through extensive simulations, we found that a value of 10 gives the best performance in our simulated scenarios. Therefore, A is set fixed as 10 in our simulations and numerical results presented in Sec. IV.

Algorithm 1 Congestion Avoidance Algorithm in High Speed Satellite Communication Networks

The congestion avoidance algorithm is triggered when *cwnd* is larger than the threshold of the slow start phase.

Initialization

1. Update the values of *RTT* according to the ACK packets received;
2. Calculate the following variables based on the values of *RTT* and link condition *p*:
   S, *AvgRTT*, *Unit_TD*, *TD*, *BaseTD*.

Congestion Windows Adjusting

1. Calculate the increment of congestion windows by Eq.(4)
2. *cwnd_new* = *cwnd_old* + *increment*.

Data Transmision

Transmit a batch of data packets according to the obtained congestion window size.

IV. SIMULATIONS AND NUMERICAL RESULTS

In order to evaluate the performance of the proposed algorithm, we have built a network topology using NS2 in which the proposed algorithm is implemented.

A. Simulation Configuration

The system topology is illustrated in Fig. 2. The satellite network provides an intermediate link to the end-to-end connection. We consider a simple bent-pipe satellite that relays packets received from the uplink to the downlink, and vice versa, without decoding the received packets. There are two gateway satellite earth stations, each at one end of the satellite link respectively. The transmission control protocol is implemented by incorporating different congestion avoidance algorithms at corresponding user terminals. The default values in NS2 for TCP configuration are used in our simulation.

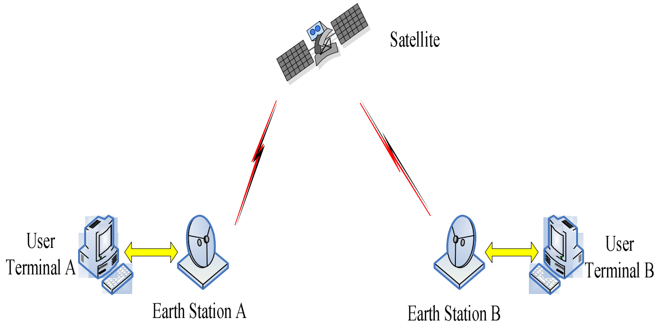

Fig. 2. System topology in simulation experiments.

In our simulations, the File Transfer Protocol (FTP) has been used as the application data source. As shown in Fig. 2, a user, terminal A, connects with earth station A and sends data to the satellite, and then the satellite relays data to user terminal B through earth station B. The two way propagation delays in our simulations have been set as constant, as 500 ms, 250 ms, and 10 ms respectively, depending on which category of satellite is used. As mentioned earlier, the total transfer delay in our networks is however a variable which depends on different transmission delay and queueing delay. The total link capacity is configured as 100 Mbps. The same as in [8], the nominal BER values of the contemporary satellite link have been configured in the range of 10^{-5}–10^{-8} for given simulation experiments, but these values could be degraded to 10^{-4} or worse. The throughput illustrated in our simulation results is the saturation throughput, which is a fundamental performance parameter, defined as the limit reached by the system throughput as the offered load increases, and represents the maximum load that the system can carry in stable conditions.

B. Numerical Results

To verify the performance of the new algorithm, we use SCPS-TP as the baseline protocol in our simulations, by only replacing the congestion avoidance algorithm with the proposed one. We refer to this new transmission control protocol using the proposed congestion avoidance algorithm as High-Speed Windows Adjusting TCP (HSWA-TCP). The simulation results on throughput comparison of these two protocols, SCPS-TP versus HSWA-TCP, are shown in Fig. 3. In this set of simulations, the GEO satellite is configured, and its corresponding two way propagation delay is set as 500 ms.

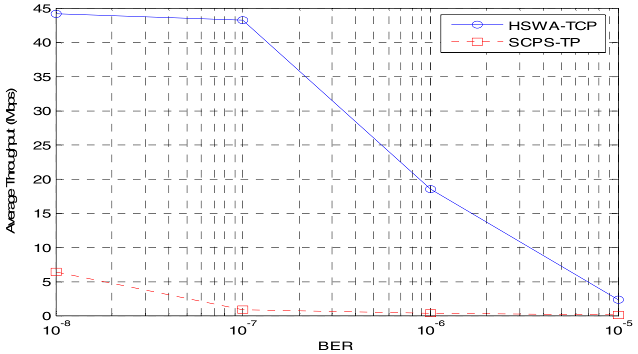

Fig. 3. TCP throughput in high speed satellite networks: SCPS-TP versus HSWA-TCP.

As illustrated in Fig. 3, the end-to-end network throughput in HSWA-TCP outperforms significantly that of the SCPS-TP, over all ranges of the simulated BER values, i.e. from 10^{-5} to 10^{-8}. In the best channel condition which is almost error-free with BER at 10^{-8}, the HSWA-TCP protocol offers approximately 7 times higher throughput than that of the SCPS-TP. More precisely, while the achieved throughput for SCPS-TP is about 6.3 Mbps, HSWA-TCP has obtained throughput at 44 Mbps. As the channel condition deteriorates, the throughput for both protocols decreases. Moreover, the performance of the HSWA-TCP protocol appears more robust under good channel conditions, but decreases faster at poorer conditions. With most BER values, the throughput in HSWA-TCP remains much higher above its counterpart, at the Mbps level until the channel condition becomes very poor. In the worst simulated case with BER at 10^{-5}, the throughput comparison becomes 2.3 Mbps for HSWA-TCP, versus 34

Kbps for SCPS-TP. That is, a huge difference of 66 times has been achieved.

The reason of this result is that HSWA-TCP could provide an efficient solution in congestion avoidance algorithm. The new congestion avoidance algorithm can quickly adapt the rate of the congestion window size and accelerate throughput according to channel conditions. When the congestion avoidance process turns into the pre-congestion sub-phase, the new algorithm still enhances the size of congestion windows in a reasonable proportional increment value. The idea of different increment corresponding to different sub-phase lets the new congestion avoidance algorithm match high speed of the satellite link, so that the benefit of the high link capacity can be sufficiently utilized. On the contrary, the increment of the congestion windows in the congestion avoidance algorithm of the SCPS-TP is always fixed in a small scale, which is obviously not responsive to the increased capacity in high speed links. This algorithm leads to the result that the satellite network throughput has been inhibited to a low level. On the other hand, we find that the HSWA-TCP throughput decays sharply with the increasing of BER values in satellite links. The reason is that, in high BER conditions, many retransmitted packets make the HSWA-TCP control process quickly turn into the congestion control phase which still uses the legacy algorithms of the SCPS-TP. The resulting throughput is however still much higher than that of the SCPS-TP case, as observed above.

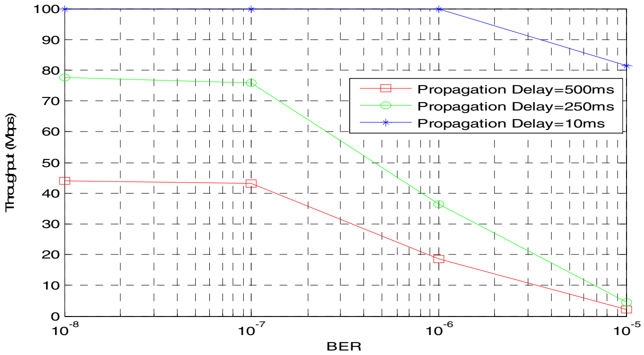

Fig. 4. The throughput performance of HSWA-TCP in different propagation delay environments.

In Fig. 4, we illustrate the throughput performance of the HSWA-TCP with different propagation delays. In general, the satellites can be categorized into GEO, Medium Earth Orbit (MEO) and Low Earth Orbit (LEO) satellites based on their altitude. The propagation delays of GEO, MEO and LEO satellites in our simulations are set by their typical values, as 500 ms, 250 ms and 10 ms respectively. Comparing the performance of HSWA-TCP with different propagation delays, we observe that higher throughput is achieved with shorter propagation delays. Especially, the throughput of HSWA-TCP reaches close to 99% of the total bandwidth of 100 Mbps, under the LEO satellite environment over a range of BER values, from 10^{-6} to 10^{-8}. In the worst investigated link case at 10^{-5} BER, the throughput performance of HSWA-TCP under LEO environment still reaches approximately 81 Mbps. In comparison, the throughput performance of HSWA-TCP under GEO and MEO environments has degraded to just 2~4 Mbps under such channel conditions. This result indicates that the HSWA-TCP algorithm is more suitable for LEO satellite communication networks with high speed links.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have firstly explored the reason why the SCPS-TP throughput is at a low level over high speed satellite links and then proposed a new TCP congestion avoidance algorithm based on the dynamic network transfer delay and link conditions in order to better accommodate the increment of the congestion window for high speed links. We demonstrated through simulations the efficiency of the proposed HSWA-TCP protocol over various channel conditions. Furthermore, we showed that, compared with the GEO and MEO environments, better throughput performance is achieved in the LEO case when using HSWA-TCP. In practice, there are also other factors that affect the TCP performance in high speed satellite networks, such as link asymmetry. How to optimize the proposed algorithm to overcome these problems is left as our future work.

REFERENCES

[1] E. Bohlin, J. C. Burgelman and C. R. Casal, "The Future of Mobile Communications in the EU", *Telematics and Informatics*,Vol. 24, No.3, pp.238-242, Aug. 2007.

[2] R. C. Durst, G. J. Miller, and E. J. Travis, "TCP Extensions for Space Communications," *Wireless Networks*, Vol.3, N. 5, pp.389-403, Oct, 1997.

[3] T. Kelly, "Scalable TCP: Improving Performance in High Speed Wide Area Networks," *Computer Communication Review*, Vol. 33, No. 2, pp. 83-91, Feb. 2003.

[4] D. X. Wei, C. Jin and S. H. Low, "FAST TCP: Motivation, Architecture, Algorithms, Performance," IEEE/ACM *Tran. on Networking*, Vol. 14, No. 6, pp.1246-1259, Dec. 2006.

[5] D. Katabi, M. Handley, and C. Rohrs, "Congestion Control for High Bandwidth-Delay Product Networks," in *Proc. ACM SIGCOMM,* June 2002, pp.19-23.

[6] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One More Bit is Enough," IEEE/ACM *Trans. on Networking*, Vol. 16, No. 6, pp.1281 – 1294, Dec. 2008.

[7] M. Gerla, M. Y. Sanadidi, R. Wang, and A. Zanella, "TCP Westwood: Congestion Window Control Using Bandwidth Estimation," in *Proc. IEEE GLOBECOM*, Nov. 2001, pp.1698-1702.

[8] F. Peng, L. Wu and C. M. Leung, "Cross-layer Enhancement of TCP Split-connections over Satellites Links," *Int. Journal of Satellite Communications and Networking*, Vol. 24, No. 5, pp.405-418, Sept. 2006.

[9] F. I. Akyildiz, G. Morabito, and S. Palazzo "TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks," IEEE/ACM *Trans. on Networking*, Vol. 9, No. 3, pp. 12-21, June 2001.

[10] K. F. Leung and K. L. Yeung, "TCP-Swift: An End-host Enhancement Scheme for TCP over Satellite IP Networks," in *Proc. IEEE ISCC*, July 2004, pp.551-555.

[11] R. C. Durst, L. G. Miller and E. J. Travis, "TCP Extension for Space Communications," *Wireless Networks*, Vol. 3, No. 5, March 1997, pp.389-403.

[12] D. T. Tran, F. J. L. Grodek, R. P. Dimond, and W. D. Ivancic., "SCPS-TP, TCP, and Rate-based Protocol Evaluation," NASA TM-2003-212001-REV1, Tech- Report, May 2005, http://gltrs.grc.nasa.gov/cgi-bin/GLTRS/browse.pl?2005/TM-2003-212001-REV1.htm

[13] W. Ruhai, H. Stephen, and C. Rama, "An Experimental Investigation of Cross-layer Optimal Packet Size in Space Internet," *Int. Journal of Satellite Communications and Networking*, Vol. 24, No. 6, November/December, 2006, p 561-577.