Advanced Transport Satellite Protocol

Muhammad Muhammad, Firat Kasmis and Tomaso De Cola
German Aerospace Center (DLR)
Institute of Communications and Navigation
82234, Wessling, Germany
Email: {Muhammad.Muhammad, Firat.Kasmis, Tomaso.DeCola}@dlr.de

***Abstract*—Mitigation of Transmission Control Protocol (TCP) performance degradation over satellite has been extensively studied over the last two decades by the scientific community, which has come up with a large set of protocol and architecture solutions. This paper proposes a novel end-to-end (E2E) transport layer protocol, namely Advanced Transport Satellite Protocol (ATSP), which is built around consolidated control theory concepts already infused in Active Queue Management (AQM) control schemes. ATSP exploits the knowledge of the bandwidth allocated to each terminal, as available from the satellite network operator. Besides, the satellite network property of being completely under control allows the joint collaboration of sender, receiver and routers in order to acquire a complete knowledge of the network status and eventually optimize the overall performance. The performance analysis shows that ATSP achieves a fair bandwidth share for all the satellite users and outperforms TCP Hybla, which is optimized for satellite links, in terms of throughput.**

I. Introduction

Transmission Control Protocol (TCP) [1] performance degradation over satellite links has been extensively studied by the scientific community over the last two decades. The main limitations of TCP reside in the congestion control mechanism therein implemented which does not allow the full satellite bandwidth exploitation because of the large bandwidth-delay product (BDP). Besides, TCP interprets segment losses as congestion signal, thus forcing the transmission rate to be reduced by halving the congestion window. In order to cope with these performance limiting factors, the scientific community has exhaustively studied the dynamics of TCP over satellite links and finally proposed a number of solutions, which aimed at improving the throughput attained on satellite links. To this end, several TCP variants like TCP Hybla [2] and TCP FAST [3] have been proposed to improve the performance of TCP in the presence of long delays and non-negligible link error ratios. Additionally, Satellite Transport Protocol (STP) [4] is another transport protocol that was designed to perform well over satellite links. Furthermore, the study of specific network architecture was also carried out in order to keep the protocol stack of end-users unchanged and to exploit the advantages of enhanced TCP versions over satellite links. This concept opened the door to the study of Performance Enhancing Proxies (PEPs) [5], possibly implemented at different layers of the protocol stack, from the data-link up to the application layer. Particularly promising is the concept of connection-splitting, which allows optimizing the performance of the transport layer protocol over the satellite link by ”splitting” the native TCP connection in the different network segments, allowing to use a specialized transport protocol over the satellite portion.

In spite of the great effort done by the scientific community in this field, no clear consensus on the most appropriate transport layer solution to be adopted has been reached yet. As a consequence, some properties of satellite networks have not been completely explored or taken into consideration in the definition of new TCP variants, thus opening the door to additional investigations and, ultimately, to the design of novel transport protocols. In particular, it is remarkable that, unlike terrestrial domains, satellite networks are a closed environment, controlled by the satellite network operator. As such, it is possible to use optimized protocols throughout the whole satellite network, by taking advantage of the status of the overall network, as available from the network operator. In other words, it is possible for satellite terminals, gateways, and end-users to jointly collaborate to optimize the overall data communication performance. To do this, it is feasible to exploit the knowledge about the bandwidth allocated to each satellite terminal and to tune the transmission rate at the transport layer based on the network congestion status computed by the other involved network entities. This can be achieved by leveraging on the control theory concepts already successfully applied to Active Queue Management (AQM) schemes and exploit the output of these queue management techniques to adaptively tune the dynamics of the transport layer protocol.

In this light, this paper proposes a novel transport protocol for satellite, called Advanced Transport Satellite Protocol (ATSP), whose building blocks are distributed amongst the network entities (sender, receiver, and router). The router implements an AQM strategy to mark packets upon imminent congestion detection; in turn, the receiver computes the network congestion status based on the number of received packets marked by the router. Finally, the sender, upon network congestion notification, tunes the rate-based mechanism to inject new packets into the network to avoid congestion losses. As a consequence, the sender is able to distinguish packet losses either due to congestion or to link errors and retransmit the missed packets without having to reduce the transmission rate, as it occurs in TCP implementations.

The work in this paper is organized as follows. In the next Section, we will present an overview about the related work in this research area. Section III will thoroughly clarify ATSP and the idea behind its operation. Simulation results and discussions will follow in Section IV. Finally, we will

978-1-4673-0921-9/12/$31.00 ©2012 IEEE

conclude in Section V.

II. Related Work

The study of TCP enhancements or design of new transport protocols has been extensively documented in the scientific literature, with particular attention to the possible counter-measures to be taken to mitigate the effect of link errors in satellite/wireless environments. In particular, the overall TCP limitations and possible mitigation in satellite scenarios are illustrated in [6] and [7]. Finally, an exhaustive survey about the main congestion control alternatives to TCP (e.g., Westwood, Hybla, etc.) worked out so far is available in [8].

On the other hand, congestion detection and avoidance have been thoroughly studied by proposing solutions implemented mostly at the network layer, building on the main concepts of control system theory. Actually, most of the network routers employ drop-tail method [9] as a simple queue management algorithm where all incoming datagrams are dropped once the queue is filled to its maximum capacity. The most critical problem is the global synchronization for all TCP streams passing through this queue, which forces all TCP connections to increase and decrease their rates simultaneously. Thus, the queues at the router will either overflow and therefore packets are dropped or be underutilized.

AQM mechanisms, aimed at controlling the rate of data packets injected into a network, have been studied and developed to avoid network congestion by monitoring the occupancy of queue buffers implemented at the network layer. Early versions of AQM algorithms, like Random Early Detection (RED) [10], address the problems of drop-tail by monitoring the average length of the queue and randomly dropping packets once this average exceeds a certain threshold, which occurs much before the buffer is full. Later on, the introduction of Explicit Congestion Notification (ECN) [11] allowed AQM algorithms to randomly mark (instead of drop) the Internet Protocol (IP) datagrams as a sign of congestion. Therefore, senders react to the marked packets by reducing their rates, i.e., as if congestion had occurred. In particular, [12] surveys the most attractive AQM techniques, giving also some insights about general congestion control concepts being applied to the Internet. As far as RED concepts are concerned, [13] provides an accurate analysis of the stability implications in TCP/IP networks, whereas the use of ECN concepts for protocol design is addressed in [14]. Cross-layer optimization and interaction between transport and network layer are instead discussed in [15]. Finally, the study of more advanced AQM schemes based on robust controller design is given in [16].

Although AQM techniques aid in minimizing the queues length at the routers and thus avoid buffer overflow, they require complex configuration. Studies [17]–[19] show that setting the parameters of these methods, especially for RED, is quite complex and usually done for a specific network setup.

To the best of the authors' knowledge, the benefits of AQM schemes and their interaction with optimized versions of TCP for satellite networks have been only partially explored. Hence, the main contribution of this paper is to propose a novel transport protocol able to exploit the satellite link bandwidth, by taking advantage of the rate allocation knowledge available from the network operator and the congestion network status information provided by an AQM strategy implemented in the router.

III. Advanced Transport Satellite Protocol

ATSP is tailored to operate over satellite links. This type of communication channels is characterized by the high bit error rate (BER) (throughout this work we will use the packet erasure rate (PER)), long round-trip time (RTT) and low bandwidth.

The corner stone of ATSP design is the knowledge of the maximum transmission rate that a sender can use. This information is available in satellite communications, because users have contracts for the maximum allowed bandwidth with the satellite operators. This permits an ATSP sender to faster utilize the expensive link bandwidth that was paid for.

Additionally, the original joint collaboration among the different network units (sender, router, and receiver) within an ATSP system allows a better estimation of the network status to perform rate control, based on information evaluated at the destination. The router in our system should be able to mark packets using the ECN field. These congestion notification marked packets are counted, in an interval, at the destination (which plays an active role within an ATSP system) and a congestion rate is estimated and fed to the sender in addition to the lost packets, if any. As a result, the source will reset its rate proportionally to the maximum allowed value and send the lost packets with the new ones. Moreover, the sender will increase its rate, for every RTT, based on the cumulative congestion information and the maximum transmission rate. In the following subsections, we will present in detail the functionality of each entity within an ATSP system.

A. Router

The router functionalities will be implemented in a satellite gateway for ATSP, where all the incoming traffic is directed to the bottleneck of the system, the satellite link. This gateway shall implement an AQM algorithm and have a limited queue size. The role of the AQM mechanism in the ATSP system is vital, since it performs the strategy of marking packets based on the way it foresees and is willing to control congestion.

As mentioned earlier, configuring RED parameters is usually very complex. Therefore, this work considers using the Proportional Integrator (PI) controller as an AQM mechanism to target a referenced queue size at the satellite gateway. Studies like [20], [21] and [22] have shown that PI controllers outperform RED mechanisms and they are easier to configure, because of their limited number of parameters. Interested readers are referred to [21] for a better understanding of the PI algorithm.

The PI controller has two main parameters, namely, the proportional gain K_p and the integral gain K_i. Additionally, there is the sampling time T.

Knowing the reference queue length q_{ref} (i.e., the setpoint of the PI controller) that the router should maintain, the queueing time for a packet in the router is $d = \frac{q_{\text{ref}}}{C}$, where C is the capacity of the bottleneck link. Therefore, in order to process all the packets, the sampling time is set to $T = \frac{d}{q_{\text{ref}}} = \frac{1}{C}$.

Further, to configure the K_p and the K_i parameters, we use the Integrated-Time-Squared-Error method (from [23]) in order to define a performance index:

$$J(K_p, K_i) = \int_{t=0}^{T} t \cdot e^2(t, K_p, K_i)\, \mathrm{d}t, \tag{1}$$

where $t \cdot e(t, K_p, K_i)$ is the weighted error of the actual and the reference queue length. Finally, what is desired is a lower performance index J, because this leads to a stable performance and to a close to setpoint output of the PI controller.

B. Receiver

An ATSP receiver plays a significant role in congestion and rate control, since it does the measurements and counts congestion notifications. It also controls the dynamics to set these values. As a result, an ATSP receiver is controlled by two variables, namely α and β:

- α is the number of packets that forces the receiver to transmit the measured congestion rate ρ. α determines the delay of the congestion notification generation;
- β is the amount of increase in the congestion rate when receiving a marked packet from the AQM. β should be very small to avoid a drastic drop in the sender's transmission window.

For our implementation we set $\alpha = 20$ and $\beta = 10^{-3}$ empirically. ρ is computed at the receiver side using:

$$\rho_{k,i} = \rho_{k,i-1} + \zeta_i \cdot \beta, \tag{2}$$

where ζ_i is the amount of the marked packets in the $i-th$ interval of α packets in the $k-th$ RTT-cycle and $\rho_{k,0} = 0$. Additionally, the destination informs the sender about the lost/dropped packets in every sent Selective Acknowledgment (SACK), which is also used for congestion notification, i.e. every α packets.

C. Sender

An ATSP sender increases and decreases its transmission rate based on the congestion status of the network that is signaled by the receiver. This is done using four variables:

- $w \doteq$ transmission window. It maintains the number of outgoing packets per RTT;
- Λ is the accumulated congestion rate, within an RTT-cycle;
- Θ is the last measured RTT;
- $W_{\max} \doteq$ maximum transmission window size, which is related to the maximum transmission rate, is set to the BDP of the satellite link and determines the maximum value of w an ATSP sender can use, which is the bandwidth the user has paid for.

First, in order for an ATSP sender to minimize burstiness and avoid buffer overflow at the router, it transmits the packets in w spaced by:

$$s_k = \frac{\Theta_k}{w_k}, \tag{3}$$

along the complete $k-th$ RTT-cycle (Θ_k).

On the other hand, the receiver will notify the sender about the instantaneous network congestion status whenever α packets are received.

1) Rate Decrease: After receiving the congestion information (in a SACK) from the destination and measuring Θ, the congestion rate Λ is incremented using:

$$\Lambda_k = \Lambda_k + \rho_{k,i}, \tag{4}$$

where $\rho_{k,i}$ is the $i-th$ received congestion rate measured from α packets in the $k-th$ RTT (check Section III-B). Then, the sender reduces its w by the received $\rho_{k,i}$ with:

$$w_k = w_{k-1} - (\rho_{k,i} \cdot w_{k-1}). \tag{5}$$

and sends the new and the lost packets (if any) in the new window w_k.

2) Rate Increase: Every RTT-cycle, an ATSP sender increases its w (thus its rate) based on the accumulative congestion rate and the maximum window size as follows:

$$w_k = \min\left\{W_{\max},\, w_{k-1} + \frac{(1-\Lambda_k)\cdot W_{\max} - w_{k-1}}{N_k}\right\}, \tag{6}$$

where $w_0 = \alpha$ (see Section III-B) and N is the slowness factor that increases upon the increase of Λ and tries to halt w from inflating. It is calculated as follows:

$$N_k = I + \Lambda_k \cdot h, \tag{7}$$

where I is the initial increase in a network with no congestion and h determines how strict the protocol reactions are upon congestion. Further, the smaller I gets, the bigger w grows in (6). From protocol experiments, we set $I = \frac{3}{2}$ and $h = 100$ to avoid bursty start-ups.

After every rate increase, Λ is set back to 0 for measuring the congestion rate of the next RTT-cycle.

IV. Simulation and Results

Our proposed protocol, ATSP, is implemented in NS-2 [24] with the purpose of testing its functionalities and foreseen goals, as well as to compare its performance to TCP Hybla, which is optimized for satellite links.

Figure 1 represents the simulation environment with N sources (S_i) and destinations (D_i), where $i = 1, 2, ..., N$, connected to the routers R1 and R2, respectively. The links between the sources and R1 and between the destinations and R2 are error free and have data rates of $100\,Mbps$ and one way delay of $5\,ms$, whereas the only bottleneck link in the scenario is between R1 and R2 with $C = 10\,Mbps$ and propagation delay of $290\,ms$. Thus, the total RTT accounts for $600\,ms$, emulating a geosynchronous (GEO) satellite link. Finally, the buffer size at R1 is set to 800 packets.

The sources are assumed to have always something to send, that is a bulk file transfer is used with unlimited size. In our simulation, we limit the number of ATSP sources (N) to 3. This is due to the time consuming computation of the K_p and K_i parameters of the PI controller. Further, in this work, we will present preliminary results, which show that our solution is a promising mechanism for satellite networks.

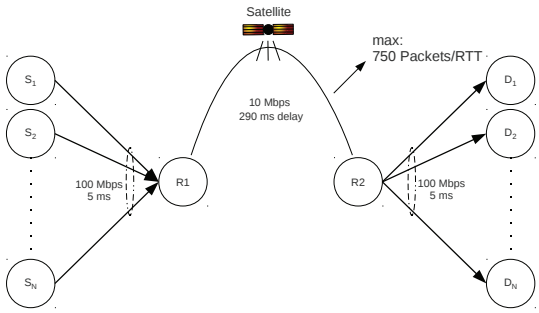


Fig. 1. Simulation environment

Equation (1) allows us to compute the performance index (J) and thus choose our desired K_p and K_i. For the PI controller, we set q_{ref} to 300 packets. Hence, since J is K_p and K_i dependent, this can be an optimization problem for multidimensional functions. Additionally, since we are searching for a minimum performance index we use the Downhill Simplex Algorithm (DSA) to find our desired values, which are $K_p = 2.607 \times 10^{-5}$ and $K_i = 2.508 \times 10^{-5}$ that correspond to $N = 3$ senders.

Within the following sections we ran multiple simulations and each simulation will reveal some properties of ATSP. In addition, the design goals are exposed as we further proceed.

A. A Closer Look at ATSP

The purpose of this simulation is to highlight some aspects of the ATSP working mechanism. We used $N = 3$ in the network topology shown in Figure 1. First, as it can be clearly seen in Figure 2(a), ATSP allows the three senders to fairly share the bandwidth of the bottleneck link as the sources are introduced into the network one after the other. More to the point, whenever a new sender starts using the network, the other senders drop their rates as fast as possible to give space for the newcomer. Please note that the total measured throughput of the 3 senders does not sum up to the bottleneck capacity ($10\,Mbps$), because of the protocols overhead, which is excluded from calculations.

B. Deactivating an ATSP Sender

In this simulation, we use the same configuration as previously, but we stop the third sender at time $t = 45\,s$, see Figure 2(b). The purpose of this test is to investigate the behavior of ATSP after providing additional capacity on the bottleneck link. As the still-alive sources are competing for the released capacity the overall throughput is dropped significantly after taking sender three out. This is due to the small amount of packets per sender injected into the network in case of congestion notification. Nevertheless, this decrease does not last long, as the other senders realize this gap and fill it as fast as possible.

C. Measuring Fairness

In order to finalize the work on ATSP running solo in the network, we run simulation of 3 senders with the optimal PI parameters and calculate the fairness index. Jain's fairness index [25] is a criterion used to determine the fair share of resources among users in a shared computer system:

$$F = \frac{\left(\sum_{i=1}^{N} x_i\right)^2}{N \cdot \sum_{i=1}^{N} x_i^2}, \tag{8}$$

where x_i is the throughput of flow i. F should lie in the interval [0, 1], where 1 means that the protocol is $100\,\%$ fair.

The fairness for 3 ATSP senders using (8) from the simulation was calculated over time. The result showed that $F \approx 1$ with very small drop to 0.9 at $t = 3\,s$ for $2\,s$.

D. Injecting Cross Traffic

In this simulation, we introduce plain Constant Bit Rate (CBR) traffic on the same bottleneck link with ATSP to monitor its performance with foreign data transport mechanism.

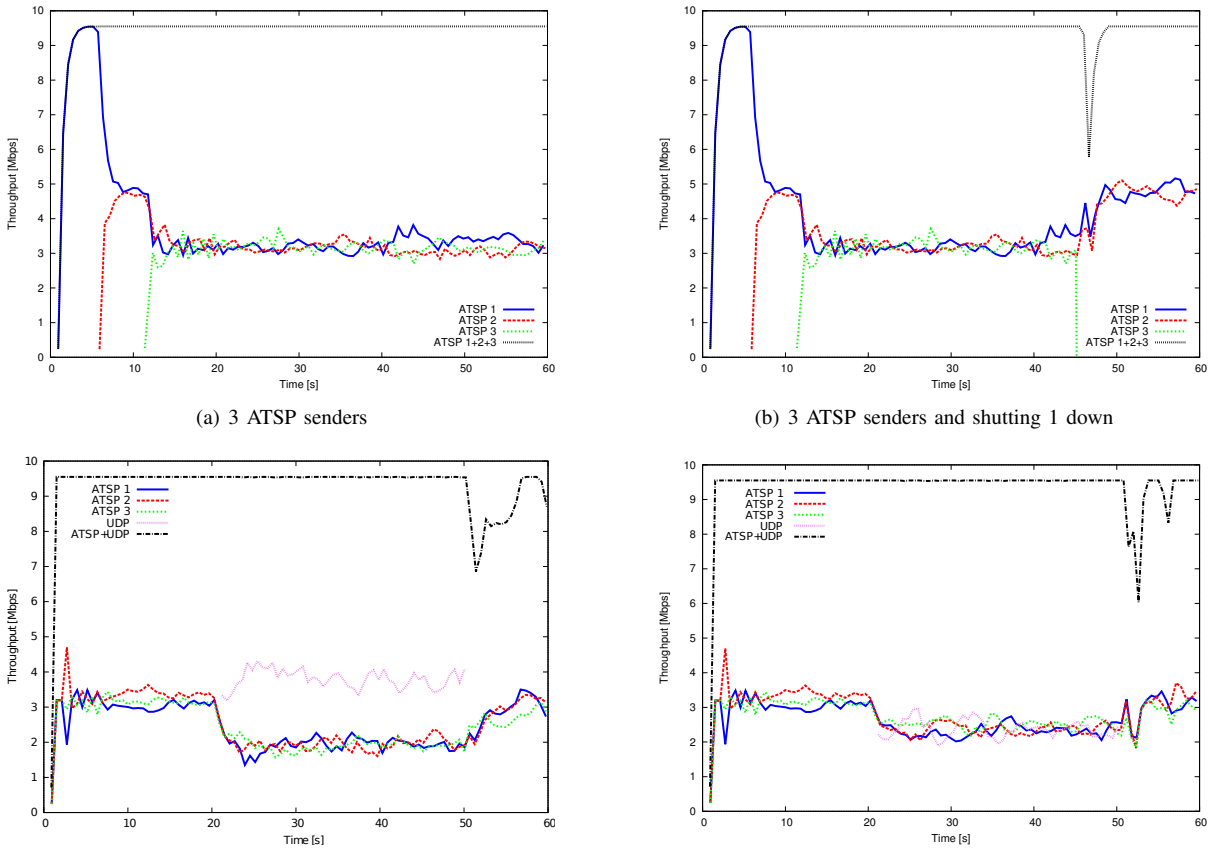
Figure 2(c) represents the case when the CBR traffic is given $4\,Mbps$ of the bottleneck link capacity. As shown, ATSP senders fairly share the remained capacity among themselves. Similarly, when the CBR traffic is allocated quarter of the bandwidth, the three ATSP senders find no problem to regulate their rates to the new situation, as seen in Figure 2(d).

E. Comparing to TCP

Due to the sensitivity of ATSP and TCP to the configuration parameters of the AQM mechanism, in our case the PI controller, it is not fair for both protocols to run together through the same AQM router. Thus, we run these simulations separately in two different networks but with the same properties and each router is configured according to the operating protocol. We use the same network topology as in Figure 1 with $N = 3$, but the senders are interchanged between TCP and ATSP with RTT set to $500\,ms$. The following simulations will compare the two protocols in two scenarios. First, we turn one of the sources off at $t = 40\,s$. Secondly, we allow random packet drop with PER value of 10^{-3} on the bottleneck link.

Since basic TCP variants are optimized to work in terrestrial networks, we choose TCP Hybla [2] that is optimized for satellite environments.

For ATSP, we used the PI controller as an AQM mechanism at R1 with the previously derived parameters. Further, we tested TCP Hybla with the router (R1) implementing drop-tail, RED, and PI techniques. The best performance of Hybla was achieved with PI with parameters from [21] as this configuration is specific for TCP flows. All buffers had a maximum capacity of 800 packets.



(a) 3 ATSP senders

(b) 3 ATSP senders and shutting 1 down

(c) 3 ATSP senders and 1 CBR flow of 4 $Mbps$

(d) 3 ATSP senders and 1 CBR flow of 3 $Mbps$

Fig. 2. Throughput of multiple ATSP senders; upper: 3 ATSP senders; lower: 3 ATSP senders with 1 CBR flow

Regarding the first simulation, the results shown in Figures 3(a) and 3(b) reflect that the Hybla version of TCP requires some time to fully utilize the released capacity after dropping out the third source. However, it is able to keep the stable synchronization of the flows. On the contrary, ATSP keeps its common behavior by trying to rapidly fill in the gap.

Finally, in the last simulation as the link drops packets (see Figures 3(c) and 3(d)), the throughput of TCP Hybla greatly degrades on link errors, whereas ATSP does not reduce its transmission rate and sends the lost datagrams that are notified in the SACK with the new ones in the next window (w). Hence, it keeps the average transmission rate stable as the throughput of TCP Hybla is reduced on average.

V. Conclusion

We presented a novel end-to-end (E2E) transport protocol for satellite links. Our new protocol, namely ATSP, takes advantage of some freely available information in a satellite network, like the maximum allowed transmission rate that permits a faster link utilization. Further, it requires all the network units to have a joint collaboration for a better network status estimation. More to the point, the sender will inject packets into the network whenever an RTT-cycle finishes. Additionally, it will decrease its rate every time a SACK is received from the destination containing congestion rate information. On the other hand, the router marks packets using AQM techniques for which an active destination counts and estimates the congestion rate that is fed back to the sender, forcing it to carefully react to congestion alerts. Finally, the simulation results showed that ATSP is able to fully utilize the bottleneck link bandwidth in short time and fairly share the capacity among its own connections also with cross traffic (CBR flows). Compared to satellite variants of TCP, like Hybla, ATSP average transmission rate is higher on average.

Acknowledgment

The authors would like to thank Dr. Matteo Berioli for his valuable time and fruitful discussions.

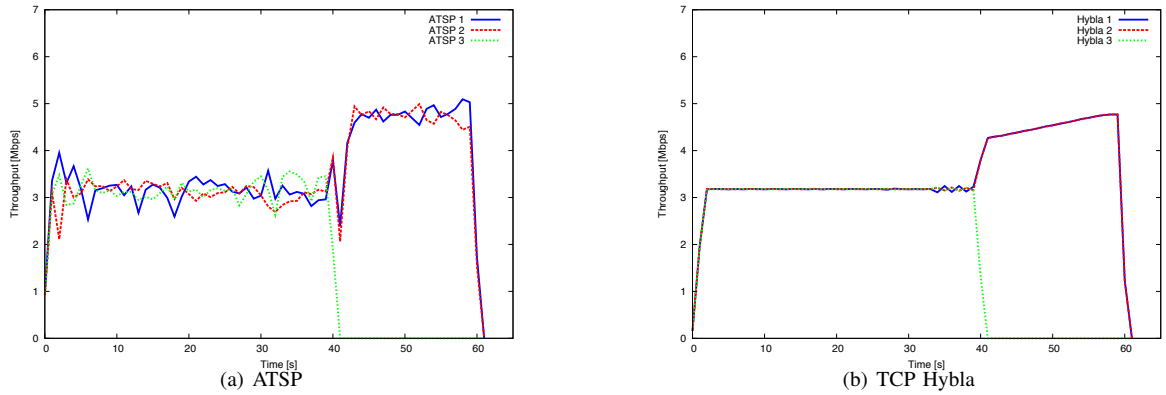
References

[1] Information Science Institute University of Southern California. (1981, Sep.) Transmission Control Protocol. IETF RFC. [Online]. Available: http://www.ietf.org/rfc/rfc793.txt

[2] C. Caini and R. Firrincieli, "TCP Hybla: a TCP Enhancement for Heterogeneous Networks," *International Journal of Satellite Communications and Networking*, vol. 22, no. 5, p. 547566, Sep. 2004.

[3] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "FAST TCP: Motivation, Architecture, Algorithms, Performance," *IEEE/ACM Transactions on Networking*, vol. 14, pp. 1246–1259, Dec. 2006. [Online]. Available: http://dx.doi.org/10.1109/TNET.2006.886335

[4] T. R. Henderson and Y. H. Katz, "Satellite Transport Protocol (STP): An SSCOP-based Transport Protocol for Datagram Satellite Networks," in *Proceedings of 2nd Workshop on Satellite-Based Information Systems*, Budapest, Hungary, Oct. 1997, pp. 23–34.



(a) ATSP (b) TCP Hybla

(c) ATSP (d) TCP Hybla

Fig. 3. Throughput comparison of ATSP and TCP Hybla; upper: shutting down a sender at $t = 40\,s$, lower: with PER of 10^{-3} on the bottleneck link

[5] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. (2001, Jun.) Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. IETF RFC. [Online]. Available: http://tools.ietf.org/html/rfc3135

[6] M. Allman, S. Dwakins, D. Glover, J. Griner, D. Tran, T. Henderson, J. Heidemann, J. Touch, H. Kruse, S. Ostermann, K. Scott, and J. Semke. (2000, Feb.) Ongoing TCP Research Related to Satellites. IETF RFC. [Online]. Available: http://tools.ietf.org/rfc/rfc2760.txt

[7] M. Allman, D. Glover, and L. Sanchez. (1999, Jan.) Enhancing TCP Over Satellite Channels using Standard Mechanisms. IETF RFC. [Online]. Available: http://www.rfc-editor.org/bcp/bcp28.txt

[8] A. Afanasyev, N. Tilley, P. Reiher, and L. Kleinrock, "Host-to-Host Congestion Control for TCP," *IEEE Communications Surveys Tutorials*, vol. 12, no. 3, pp. 304 –342, May 2010.

[9] D. E. Comer, *Internetworking with TCP/IP, Volume 1: Principles, Protocols and Architectures*, 4th ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2000.

[10] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transanctions on Networking*, vol. 1, pp. 397–413, August 1993. [Online]. Available: http://dx.doi.org/10.1109/90.251892

[11] K. Ramakrishnan, S. Floyd, and D. Black. (2001, Sep.) The Addition of Explicit Congestion Notification (ECN) to IP. IETF RFC. [Online]. Available: http://tools.ietf.org/rfc/rfc3168.txt

[12] S. Ryu, C. Rump, and C. Qiao, "Advances in Internet Congestion Control," *IEEE Communications Surveys Tutorials*, vol. 5, no. 1, pp. 28–39, 3rd. Quarter 2003.

[13] L. Tan, W. Zhang, G. Peng, and G. Chen, "Stability of TCP/RED Systems in AQM Routers," *IEEE Transactions on Automatic Control*, vol. 51, no. 8, pp. 1393–1398, Aug. 2006.

[14] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One More Bit is Enough," *IEEE/ACM Transactions on Networking*, vol. 16, no. 6, pp. 1281 –1294, dec. 2008.

[15] J. Wang, L. Li, S. Low, and J. Doyle, "Cross-layer Optimization in TCP/IP Networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 3, pp. 582 – 595, Jun. 2005.

[16] Q. Chen and O. W. W. Yang, "Robust Controller Design for AQM Router," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 938 –943, May 2007.

[17] W.-C. Feng, D. Kandlur, D. Saha, and K. Shin, "A Self-Configuring RED Gateway," in *IEEE Proceedings of INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, New York, NY, USA, Mar. 1999, pp. 1320 –1328.

[18] D. Lin and R. Morris, "Dynamics of Random Early Detection," *SIGCOMM Computer Communications Rev.*, vol. 27, pp. 127–137, Oct. 1997. [Online]. Available: http://doi.acm.org/10.1145/263109.263154

[19] V. Firoiu and M. Borden, "A Study of Active Queue Management for Congestion Control," in *IEEE Proceedings of INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, Tel Aviv, Israel, Mar. 2000, pp. 1435 –1444.

[20] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "A Control Theoretic Analysis of RED," in *IEEE Proceedings of INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, Anchorage, AK , USA, Apr. 2001, pp. 1510 –1519.

[21] ——, "On Designing Improved Controllers for AQM Routers Supporting TCP Flows," in *IEEE Proceedings of INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, Anchorage, AK , USA, Apr. 2001, pp. 1726 –1734.

[22] ——, "Analysis and Design of Controllers for AQM Routers Supporting TCP Flows," *IEEE Transactions on Automatic Control*, vol. 47, no. 6, pp. 945 –959, Jun. 2002.

[23] N. Killingsworth and M. Krstic, "PID Tuning Using Extremum Seeking: Online, Model-free Performance Optimization," *Control Systems, IEEE*, vol. 26, no. 1, pp. 70 – 79, feb. 2006.

[24] "The Network Simulator NS-2." [Online]. Available: http://www.isi.edu/nsnam/ns/

[25] R. Jain, W. Hawe, and D. Chiu, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems," DEC Research Report TR-301, Tech. Rep., 1984.