A Recursive, Explicit and Fair Method to Efficiently and Fairly Adjust TCP Windows in Satellite Networks

Tarik Taleb, Nei Kato and Yoshiaki Nemoto
Graduate School of Information Sciences - Tohoku University
Sendai, Japan 980–8579
Email: taleb,kato,nemoto@nemoto.ecei.tohoku.ac.jp

***Abstract*— As originally specified, TCP did not perform well over satellite network systems, systems known with their rapidly time-varying network topologies. This paper addresses some vexing attributes that impair TCP performance in LEO satellite networks. The paper proposes a scheme that allows satellite systems to automatically adapt to the number of active TCP flows, the free buffer size and the bandwidth-delay product of the network. The proposed scheme controls the efficiency of the system by matching the aggregate traffic rate to the sum of the link capacity and total buffer size. This attribute helps to adjust TCP's aggressiveness and to prevent persistent queues from forming. The system min-max fairness is achieved by allocating bandwidth among individual flows in proportion with their RTTs.**

Simulation results elucidate that the proposed scheme substantially improves the system fairness, reduces the number of packet drops and makes better utilization of the bottleneck link. The results demonstrate also that the proposed scheme works properly in more complicated environments where connections traverse multiple bottlenecks and the available bandwidth may change over data transmission time.

I. Introduction

The need for satellite communication systems has grown rapidly during the last few years. For more than three decades, GEO satellite systems have been successful in providing some commercial services. However, requirements of lower propagation delays, in conjunction with the coverage of high latitude regions, have sparked the development of new satellite communication systems called Low Earth Orbit (LEO) satellite systems.

Communication over LEO satellite networks exhibits different characteristics compared to that over the traditional satellite or wired networks. The success of LEO satellite networks in delivering high-speed access hinges on the ability of the underlying protocols to function correctly and efficiently in LEO systems. The effect of such a communication environment on the working of Transmission Control Protocol (TCP), the backbone of today's Internet protocol communication [1], underpins the research work outlined in this paper. The paper firstly addresses some vexing attributes that impair TCP performance in LEO satellite systems. It next determines the appropriate modifications to help TCP suite well in such environments.

TCP usually results in drastically unfair bandwidth allocations when multiple connections share a bottleneck link. This unfairness issue appears when the number of flows varies over time, as has been indicated in studies of terrestrial wide-area network traffic patterns [2]. In case of multi-hops satellite constellations, the unfairness issue becomes more substantial. Indeed, when a group of users, with considerably different RTTs, compete for the same bottleneck capacity, TCP's undesirable bias against long RTT flows becomes more significant, as is reported in [3]. Additionally, satellite networks are well characterized by frequent handover occurrences [4]. A handover occurrence may force a TCP sender to either suddenly alter its path and compete for bandwidth with a different group of connections, or just keep its path but be sharing the same link with newly coming connections. Both cases will eventually result in an abrupt change in both the flows count and the connection's RTT. If all TCP senders keep sending data without any adjustment in their sending rates, an increase in the flows count will result in overloading the link with data packets causing excessive queuing delays, large number of packet drops and throughput degradation, whereas a decrease in the flows count will lead to a waste of bandwidth and ultimately lower link utilization.

This paper argues that the TCP rate of each flow should be adaptively adjusted to the available bandwidth when the number of flows that are competing for a single link, changes over time. An explicit and fair scheme is developed. The key concept of the proposed scheme is to match the aggregate window size of all active TCP flows to the effective network bandwidth-delay product. At the same time, the scheme provides all the active connections with feedbacks proportional to their RTT values so that the system converges to optimal efficiency and max-min fairness. Feedbacks are signaled to TCP sources through the receiver's advertised window (RWND) field in the TCP header of acknowledgments. Senders should accordingly regulate their sending rates. This operation can be accomplished without changing the protocol. As a result, it requires no modification to the TCP implementation in the terminals. The proposed scheme is dubbed *Recursive, eXplicit and Fair Window Adjustment (RXFWA)*.

The remainder of this paper is structured as follows. Section II highlights the relevance of this work to the state-of-art in the context of TCP performance over satellite networks. The key design philosophy and distinct features that were incorporated in the proposed scheme are described in Section III. Section

IV portrays the simulation environment and discusses the simulation results. Following this, the paper concludes in Section V with a summary recapping the main advantages and achievements of the proposed scheme.

II. State-of-Art Related Work

Despite the widespread acceptance of TCP/IP in terrestrial networks, its performance over satellite networks is still limited [5]. To diminish the negative impact slow start has on performance, the congestion window is initially set to a value larger than 1 packet size but lower than 4 packets size [6]. [7] has investigated the usage of low-priority dummy segments to probe the availability of network resources. In TCP/SPAND [8], network congestion information is cached at a gateway and shared among many co-located hosts. Using this congestion information, TCP senders can make an estimate of the optimal initial congestion window size at both connection start up and restart after an idle time. Other researchers have discussed the potential of a technique called TCP Spoofing [9] [10]. In this technique, a router near the TCP sender prematurely acknowledges TCP segments destined for the satellite host. This operation gives the source the illusion of a short delay path speeding up the sender's data transmission. One more similar concept is TCP Splitting where a TCP connection is split into multiple connections with shorter propagation delays [11]. All in all, most proposed solutions are algorithms pertaining to only efficiency issues, mainly problems related to slow-start phase. Whereas, TCP behavior under many competing flows in satellite networks has not been sufficiently explored.

Current TCP implementations infer the congestion state of a network only from implicit signals such as arrival of ACKs, timeouts and receipt of duplicate acknowledgments (dupACKs). In the absence of such signals, the TCP congestion window grows up to the maximum socket buffer advertised by the receiver. In case of multiple flows competing for the capacity of a given link, this additive increase policy will cause severe congestion, degraded throughput and unfairness.

One approach to control congestion is to employ scheduling mechanisms, fair queuing and intelligent packet-discard policies such as Random Early Marking (REM) [12] and Random Early Discard (RED) [13] combined with Explicit Congestion Notification (ECN) [14]. These policies require a packet loss to signal early network congestion to TCP sources. However, in case of large delay links (e.g. satellite links), these policies become inefficient and may still cause timeouts. By the time the source starts decreasing its sending rate because of a packet loss, the network may have already been overly congested. Low et al. [15] have shown through mathematical analysis the inefficiency of these Active Queue Management schemes (AQM) in environments with high bandwidth-delay product such as satellite networks.

AQM limitations can be mitigated by addition of some new mechanisms to the routers to complement the endpoint congestion avoidance policy. These mechanisms should allow network elements between a TCP source and a TCP destination to acknowledge the source with its optimal sending rate. By so doing, the whole system becomes self-adaptive to traffic demands and more active in controlling congestion and buffer overflows. To cope with TCP limitations in high bandwidth-delay product networks, several studies have been conducted providing valuable insight into TCP dynamics in such environments. Katabi et al. [16] proposed a new congestion control scheme, eXplicit Control Protocol (XCP). The scheme substantially outperforms TCP in terms of efficiency in high bandwidth-delay product environments. However, the main drawback of this protocol is that it assumes a pure XCP network and requires significant modifications at the end-system. Explicit Window Adaptation (EWA) [17] and WINdow TRAcking and Computation (WINTRAC) [18] suggest an explicit congestion control scheme of the window size of TCP connections as a function of the free buffer value. Since the computed feedback is a function of only buffer occupancy and does not take into account link delay or link bandwidth, the two schemes are likely to run into difficulty in face of high bandwidth or large delay links similarly to AQMs. Additionally, EWA and WINTRAC achieve fairness only in the distribution of the maximum achievable window size. The two schemes remain grossly unfair towards connections with high variance in their RTTs distribution.

In the sphere of TCP behavior under many competing flows in satellite networks, [19] and [20] appear to be the first work to introduce the idea of explicit signaling to improve both efficiency and fairness of TCP in broadband satellite networks. Simulation results reveal the good performance of the proposed scheme in terms of both system efficiency and fairness. However, the conducted simulations considered only a simple network environment. It was assumed that only one-bottleneck was traversed by many connections and that all flows were always making full use of their allocated windows. It was also assumed that the bandwidth available to TCP flows remained steady over time. These assumptions are, however, not generally valid. For instance, when there are multiple bottlenecks, a TCP connection may send data with rates fed back by a given bottleneck but smaller than the windows allocated by the other bottlenecks. This will obviously cause the underutilization of some of the links. In addition, due to the variability of higher priority traffic, the available bandwidth is not always steady and may change over data transmission time. These kind of situations may attenuate the performance of the scheme.

III. Operational Overview of the Proposed Scheme, RXFWA

Before delving into details of the feedback computation method, first is a description of how the scheme exploits some features of satellite networks to make an approximate estimate of flows RTT and the bandwidth-delay product of the network.

Prior knowledge of the RTT estimates is usually not available at network elements in terrestrial networks. However, in case of multi-hops satellite constellations, flows RTT can be handled by a simple monitoring of hops count in the backward and forward traffic of each flow. Hops count of each flow can

TABLE I
FLOWS GROUP STATE TABLE

Group ID	RTT	Flows Count	Flows ID	Last Packet Transmission Time
1	RTT_1	n_1	1.1 ⋮ 1.n_1	$t_{1.1}$ ⋮ $t_{1.n_1}$
2	RTT_2	n_2	2.1 ⋮ 2.n_2	$t_{2.1}$ ⋮ $t_{2.n_2}$
⋮	⋮	⋮	⋮	⋮
M	RTT_M	n_M	M.1 ⋮ M.n_M	$t_{M.1}$ ⋮ $t_{M.n_M}$

be easily computed from Time to Live (TTL) field in the IP header of both ACK and data packets. Let H_b and H_f denote the number of hops traversed by an acknowledgment packet in the backward traffic and the number of hops traversed by a data packet in the forward traffic before entering the router in question, respectively. Since queuing delays have minimal contribution in the one-way propagation delay and the value of the inter-satellite link delay remains constant in multi-hops satellite networks [21], the estimated value of the connection RTT can be expressed as:

$$RTT = 2 \cdot (H_b + H_f + 2) \cdot ISL_{delay} \quad (1)$$

where ISL_{delay} denotes the inter-satellite link delay.

Using knowledge of the RTT estimates, flows are grouped according to the number of traversed hops, as shown in table I. Each group of flows is identified by a group ID. At each router, a group G is defined as the set of flows that traverse the same number of hops and thus have equal RTTs, RTT_G. Flows are identified also by a flow ID and are defined as streams of packets sharing the quintuple: source and destination addresses, source and destination port numbers, and protocol field. A flow is considered to be in progress if the time elapsed since its last packet transmission time is inferior than a predetermined threshold δ. This threshold is always updated to the most recent estimate of the average RTT of all active flows traversing the router. The router's most recent estimate of the average RTT is referred to as RTT_{avg} throughout this paper.

At time $t = n \cdot RTT_{avg}$, the feedback value of flows that belong to the κ^{th} group, $F_\kappa(n)$, is computed as

$$F_\kappa(n) = \frac{RTT_\kappa}{\sum_{j=1}^{M} n_j \cdot RTT_j} \cdot \Upsilon(n) \quad (2)$$

$$\Upsilon(n) = \Upsilon(n-1) + \phi\Big(Bw \cdot RTT_{avg} - \Upsilon(n-1)\Big) + \psi\Big(B - Q(n-1)\Big)$$

where M, B and Bw are the total number of flows groups, router's buffer size and the link bandwidth, respectively. n_j and RTT_j denote the size of the j^{th} group and the RTT value of its flows, respectively. $\Upsilon(n)$ and $Q(n)$ denote the aggregate TCP window size and the router's queue occupancy at time $t = n.RTT_{avg}$. ϕ and ψ are constant parameters. It should be recognized that ϕ and ψ play a significant role in exploiting well the system spare bandwidth and free buffer size, respectively. In other words, in case of optimum values of ϕ and ψ, when some connections are not making full use of their allocated bandwidths during the interval time $[(n-1)RTT_{avg}, nRTT_{avg}]$, the spare bandwidth, $(Bw \cdot RTT_{avg} - \Upsilon(n-1))$, will increase and the buffer occupancy, $Q(n-1)$, will go down causing an increase in the computed feedbacks of the other connections at time $t = n \cdot RTT_{avg}$. This will help to fully utilize the link capacity while maintaining small buffer sizes. Notice that $\Upsilon(n)$ is computed in a recursive manner, hence the name of the proposed scheme, *Recursive, eXplicit and Fair Window Adjustment (RXFWA)*.

The window feedback is computed every RTT_{avg} time and is written in the receiver's advertised window (RWND) field carried by the TCP header of acknowledgment packets. The RWND value can only be downgraded. If the original value of the receiver's advertised window, which is set by the TCP receiver, exceeds the feedback value computed by a downstream node, the receiver's advertised window is reduced then to the computed feedback value. A more congested router later in the path can further mark down the feedback by overwriting the receiver's advertised window field. Ultimately, the RWND field will contain the optimal feedback from the bottleneck along the path. When the feedback reaches the sender, the sender should react to the message and accordingly updates its current window. This dynamic adjustment of RWND value will help to smooth the TCP burstiness and to achieve efficiency and a fair window size distribution among all competing flows.

IV. SIMULATION SET-UP AND PERFORMANCE EVALUATION

The design of the simulation setup relies on Network Simulator (NS) [22]. In all simulations, TCP sources implement the TCP NewReno version [23]. The data packet size is fixed to $1kB$. In order to remove limitations due to small buffer sizes on the network congestion, buffers equal to the bandwidth-delay product of the bottleneck link are used [24]. All routers use Drop-Tail as their packet-discarding policy. Simulations were all run for $20s$, a duration long enough to ensure that the system has reached a consistent behavior. The inter-satellite link delay, ISL_{delay}, is set to $20ms$ and all up-links and down-links are given a capacity equal to $10Mbps$. All links are presumed to be error-free. This assumption is made so as to avoid any possible confusion between throughput degradation due to packet drops and that due to satellite channel errors.

A. Effect of ϕ and ψ

The aim of this section is to determine how closely ϕ and ψ correlate to each other; specifically, to examine whether one factor weighs more than the other and if so, how a compromise can actually be formed between the two variables. In this experiment, the used network configuration is modeled as a single network bottleneck (Fig. 1-(a)). The bottleneck link is composed of three satellites. The inter-satellite link

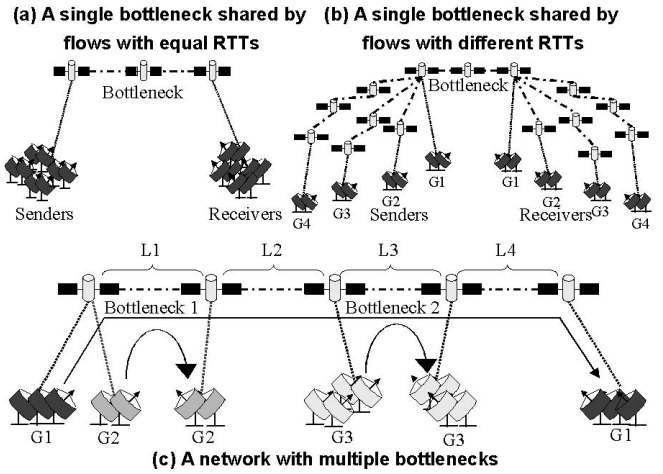


Fig. 1. Simulation Environments

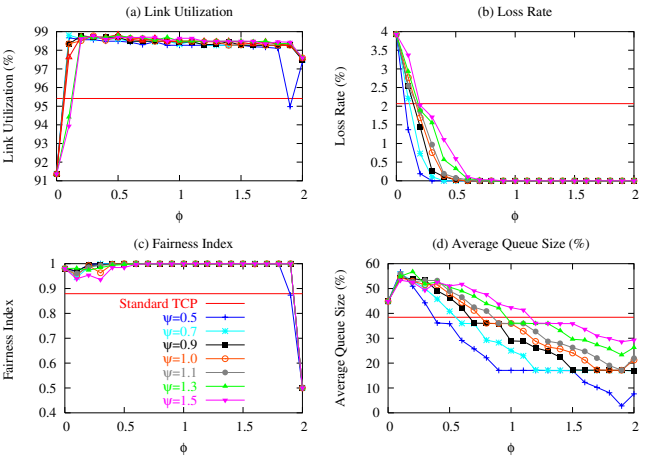


Fig. 2. Effect of ϕ and ψ on the system overall performance

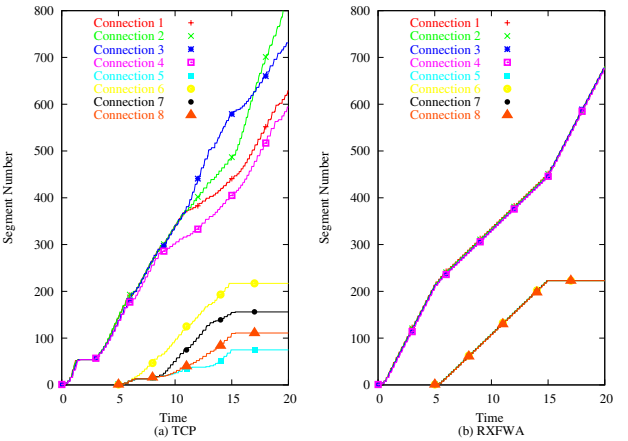


Fig. 3. Sequence Number Growth (Dynamic Environment)

bandwidth is set to $1.544Mbps$ (e.g. T1). 8 connections are activated at the beginning of the simulation. Their starting time is uniformly distributed from 0 to $5ms$ to avoid bursty losses at the simulation launch time. All connections remain open for $20s$, the simulation running time. Fig. 2 graphs the overall network performance in terms of link utilization, drops rate, fairness index [1], and average queue size for different values of ϕ and ψ. Simulation results show that large values of ψ cause higher packet losses and larger queue sizes. This can be explained by the fact that large values of ψ cause large bursts in the network which obviously lead to larger queue sizes and ultimately higher number of packet drops. The simulation results indicate also that the system experiences significantly higher number of drops and larger queue size in case of small values of ϕ. Notice also that the system tends to be more unfair and the overall throughput gets degraded when ϕ takes values in the vicinity of 0. The main reason behind this performance is that in case of small values of ϕ, the effect of ϕ in the feedback computation becomes minimal and the proposed scheme becomes, in nature, similar to the standard TCP. On the other hand, large values of ϕ reduce the average queue size but degrade the system throughput and considerably affect its fairness.

Being interested in setting ϕ and ψ to fixed values so no further tuning of the system is required, we develop a mathematical model in Appendix I. The model helps to find the optimal values of ϕ and ψ that guarantee the system's stability. In the remainder of this paper, ϕ and ψ are set to 1.25 and 0.75, respectively.

B. RXFWA's Robustness in Dynamic Environments

This experiment aims to illustrate the effect of the change in flows count on TCP behavior and to examine how RXFWA scheme adapts to sudden increases or decreases in traffic demands. Similarly to the previous experiment, a symmetric and simple network bottleneck shared by 8 connections is considered (Fig. 1-(a)). As for the change in traffic demands, we start the simulation with half of the flows (1^{st} till 4^{th}) that are let active for $5s$, when the 2^{nd} half of flows (5^{th} till 8^{th}) are launched. At time $t = 15s$, the last half of flows is deactivated. The remaining connections are left active until the end of the simulation.

Fig. 3 illustrates the sequence number growth of each simulated TCP connection. The RXFWA scheme significantly outperforms TCP: the TCP segment number of all TCP connections progresses evenly when RXFWA is used. With RXFWA, the slope of the sequence number lines decreases at time $t = 5s$ as 4 new flows enter the system, and increases at time $t = 15s$ as a result of extra bandwidth becoming available for the remaining active connections. Observe how RXFWA helps all flows to progress fairly and to behave in an identical way, whereas, in case of only standard TCP, TCP flows exhibit great deviation. Notice also that, without the RXFWA, the slope of the sequence number growth of the newly entering flows exhibits some small oscillations or remains steady most probably due to timeouts. However, with

[1]The fairness index involves the relative throughput of flows sharing a link defined as:

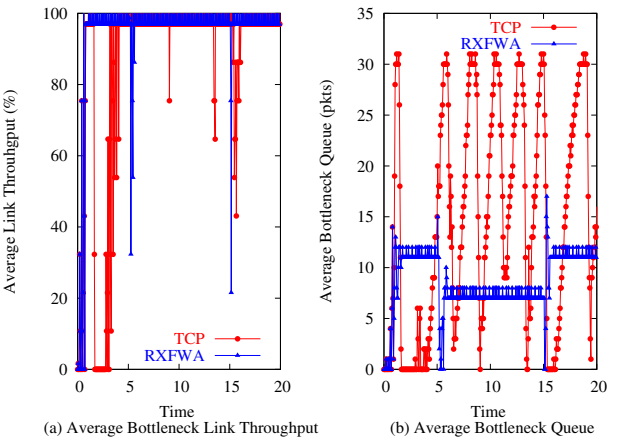
$$F(x) = \frac{(\sum_{i=1}^{N} x_i)^2}{N \cdot \sum_{i=1}^{N} (x_i)^2}$$

where N and x_i denote the total number of flows and the actual throughput of the i^{th} flow, respectively.

RXFWA, it is observed that none of the simulated flows experienced a timeout, and, that the sequence number growth of all flows is parallel. The underlying reason for this performance consists in the RXFWA's ability to rapidly adapt to changing network conditions and to fairly divide the available bandwidth among the competing flows. When new connections enter the network, the RXFWA scheme quickly computes a smaller window feedback and when some connections exit the network, it feeds back TCP sources with larger window sizes. Additionally, the RXFWA scheme does not allow flows to obtain portions of the available bandwidth larger than their fair shares. Consequently, when a flow gets some of its packets dropped, the flow will always be guaranteed a fair portion of the available bandwidth. The flow will then use the allocated portion to recover from losses without entering the slow-start phase that can be triggered if timeouts occur. This assures a fair progression in window size for both newly coming and already existing flows.

The RXFWA outperforms standard TCP not only in terms of higher fairness and lower packet drops, but also in improving the link utilization and minimizing the queue size. The queue occupancy of the bottleneck link and the average throughput, measured in intervals of $100ms$, are presented in Fig. 4. The figure indicates that without RXFWA, the throughput fluctuates irregularly. Without the RXFWA scheme, TCP sources constantly probe for available bandwidth in the network until there is no space in the pipe to maintain new packets. At that time, drops become inevitable. The throughput loss is mainly due to the synchronization of these packet losses and their simultaneous recovery. These synchronized losses cause the connections to simultaneously reduce their windows and ultimately result in degraded throughput. In contrast, RXFWA's utilization is always near the total capacity of the link and exhibits very limited oscillations [2].

Fig. 4-(b) demonstrates that without RXFWA, TCP senders increase their window size until they cause buffer overflows. This cycle occurs repeatedly and causes the queue size to oscillate more frequently. Changes in traffic demands result also in transient overshoots in the queue size. These transient overshoots and the large oscillations in the queue size can cause timeouts and frequent underflows, thereby resulting in a substantial idling of the bottleneck link. On the other hand, in the case of the RXFWA scheme, the buffer underflows that are mostly due to the change in traffic demands are brief and do not significantly affect the bottleneck link utilization. While RXFWA aims to reduce queue sizes to minimum by preventing persistent queues from forming, it is observed that the obtained bottleneck queue size remains constant and is larger than a certain number of packets. This is mainly due to the simultaneous release of entire windows of packets, in a single burst, at the beginning of each RTT. One possible solution to this issue is the transmission of packets in a steady stream (multiple, small bursts) over the entire course of the RTT [25].



(a) Average Bottleneck Link Throughput (b) Average Bottleneck Queue

Fig. 4. Average Link Utilization and Queue Size Variations in a Dynamic Environment

C. System Fairness

To explore the performance of RXFWA in environments with high variance in the RTT distribution, the network topology depicted in Fig. 1-(b) is considered. All flows are grouped in four equally-sized groups. Flows belonging to the i^{th} group traverse $2i+1$ satellites causing each flow to have an RTT of $((2i+2)\cdot 40ms)$. A number of test scenarios was created by setting the Inter-Satellite Link bandwidth to different typical link speeds and fixing a maximum number of flows [3], MAX, for each link type: $1.5Mbps$ (e.g. T1, $MAX = 8$), $10Mbps$ (e.g. T2, $MAX = 32$), $45Mbps$ (e.g. T3, $MAX = 128$) and $155Mpbs$ (e.g. OC3, $MAX = 512$). All connections are activated at the beginning of the simulation and remain open during the simulation running time. Their starting time is uniformly distributed from 0 to $5ms$ to avoid bursty losses at the simulation launch time.

The individual flow throughputs of the four scenarios are plotted in Fig.5. In case of the RXFWA scheme, the figure indicates that all flows could send nearly the same amount of packets. The figure demonstrates also the greediness of standard TCP: short RTT connections are seen to conquer most of the link bandwidth and send significantly larger number of packets compared to the long RTT connections. The obtained results confirm as well that the RXFWA estimate of the average RTT of the system operates correctly in environments with high variance in the RTT distribution.

D. Performance in Presence of Multiple Bottlenecks

So far, a single bottleneck environment has been considered. The remainder of this section envisions a general case where connections traverse multiple bottlenecks. Fig.1-(c) shows the example network configuration used in this study. The abstract configuration consists of three groups of flows. Group 1 consists of three flows and shares bandwidth of the first bottleneck (link $L1$) with the two flows of group 2. Group 3 is formed of four flows and shares bandwidth of the second

[2]Except at time $t = 0s$ and $t = 5s$ due to the entry of many connections all at the same time.

[3]So that a minimum value of link fair-share can be guaranteed among all competing flows.

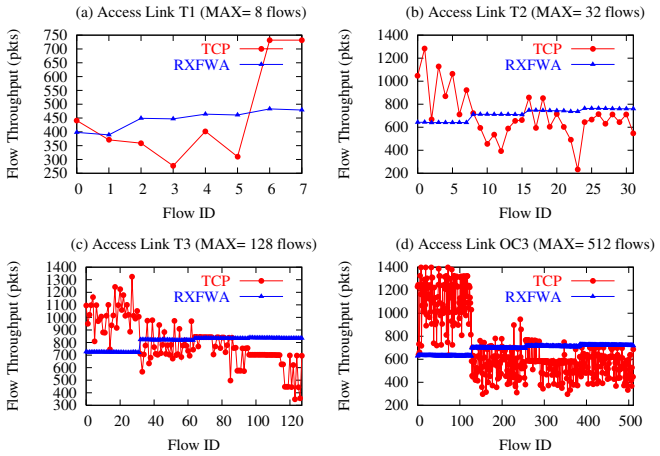


Fig. 5. Individual Flow Throughputs for Different Access Link Types

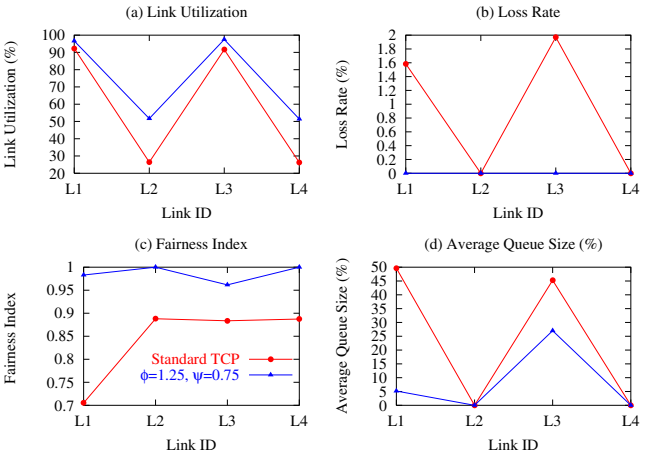


Fig. 6. Overall Performance for Each Link

bottleneck (link $L3$) with flows of group 1. Inter-satellite links are given a capacity of $1.544Mbps$ (e.g. T1).

Fig.6 graphs the link utilization, drops rate, fairness index and average queue size experienced by the four simulated inter-satellite links. As explained earlier, the feedback value of a connection can be only downgraded along its path. If the feedback value set by a node exceeds the value computed by its downstream counterpart, the feedback value is marked down to the smaller value. In case of the considered network topology, TCP connections of group 1 are assumed to send data with rates fed back by link $L3$ that are smaller than the windows allocated by link $L1$. This compels flows of group 1 not to fully utilize their allocated bandwidths at link $L1$ causing the spare bandwidth to increase and the buffer occupancy to go down. This eventually leads to an increase in the computed feedbacks at link $L1$. Flows of group 2 will transmit data at rates equal to the computed feedback and this will help to fully utilize the $L1$ link capacity while maintaining small buffer sizes. Results of Fig.6 demonstrate the high utilization of both links $L1$ and $L2$ and confirm this observation. It is noticed that without RXFWA, link $L1$ is also highly utilized but this is at the price of lower system fairness. This is likely because flows of group 2 conquer most of the link bandwidth given their short RTTs. Observe also the high number of losses and large queue sizes experienced when TCP connections are controlled by only standard TCP.

V. Conclusion

In this paper, we proposed a TCP window adjustment method specifically designed for multi-hops LEO satellite constellations. The proposed scheme exploits some features of multi-hops satellite networks to make an approximate estimate of flows RTT and the bandwidth-delay product of the network. Using these two estimates, the scheme matches the sum of window sizes of all active TCP connections sharing a bottleneck link to the effective bandwidth-delay product of the network. The proposed scheme improves the system min-max fairness by assigning for each flow a weight proportional to its RTT.

Extensive simulation results demonstrate that RXFWA has the potential of substantially improving the system fairness, reducing the number of losses and making better utilization of the link. Experiments with dynamic changes in traffic demands showed that RXFWA managed to control well the buffer occupancy and to achieve stability when a change in traffic load occurred. A multiple-bottlenecks network was considered and the good performance of RXFWA in such environment was verified by simulation results. As for the feedback computation, it is fairly simple and requires only a few additions and a few multiplications every RTT_{avg} interval of time. Due to paper length limitation, the authors are compelled to omit discussion on the necessary computation load. The interested reader is directed to [20].

Appendix I: Stability Analysis of the Feedback Control Scheme

This appendix discusses the stability of the proposed scheme through a mathematical analysis. From equation 2, the aggregate TCP window size at time $(t = n \cdot RTT_{avg})$ is expressed as

$$\begin{aligned}\Upsilon(n) &= \Upsilon(n-1) + \phi\Big(Bw \cdot RTT_{avg} - \Upsilon(n-1)\Big) + \\ &\quad \psi\Big(B - Q(n-1)\Big).\end{aligned}$$

Noticing that $Q(n-1)$ is the queue size resulted from unsent packets of windows $W(n-2)$, $Q(n-1)$ can be expressed as follows

$$\begin{aligned}Q(n-1) &= \sum W(n-2) - Bw \cdot RTT_{avg} \\ &= \Upsilon(n-2) - Bw \cdot RTT_{avg}.\end{aligned}$$

We then obtain the following system-control equation

$$\Upsilon(n) + \Omega_1 \cdot \Upsilon(n-1) + \Omega_2 \cdot \Upsilon(n-2) = \Omega_3 \qquad (3)$$

where,

$$\begin{aligned}\Omega_1 &= \phi - 1 \\ \Omega_2 &= \psi \\ \Omega_3 &= \psi \cdot B + Bw \cdot RTT_{avg} \cdot (\phi + \psi).\end{aligned}$$

The equation can be simplified to

$$X(n) + \Omega_1 X(n-1) + \Omega_2 X(n-2) = 1$$

where,

$$X(n) = \Upsilon(n) + \frac{\Omega_3}{1 + \Omega_1 + \Omega_2}.$$

Using the z-transform method, the discrete open-loop transfer function of the system is as follows

$$\begin{aligned} Z\{X(n)\} &= X_z(z) \\ X_z(z) &= \frac{z^3}{z-1} \cdot \frac{1}{z^2 + \Omega_1 z + \Omega_2}. \end{aligned}$$

Using Tustin's approximation

$$X_z(z) = X_s(s)_{s=\frac{2}{RTT_{avg}} \cdot \frac{1-z^{-1}}{1+z^{-1}}}$$

we obtain the continuous open-loop transfer function

$$X_s(s) = M_1 \cdot \frac{(s + M_2)^3}{s(M_3 s^2 + M_4 s + M_5)}$$

where,

$$\begin{aligned} M_1 &= \frac{RTT_{avg}^2}{2} \\ M_2 &= \frac{2}{RTT_{avg}} \\ M_3 &= (\Omega_2 - \Omega_1 + 1) RTT_{avg}^2 \\ M_4 &= 4(1 - \Omega_2) RTT_{avg} \\ M_5 &= 4(\Omega_1 + \Omega_2 + 1). \end{aligned}$$

The system is stable if all the roots of the transfer function denominator polynomial have negative real parts. This condition is satisfied when

$$\begin{aligned} 0 \le \Omega_1 &\le \Omega_2 + 1 \\ \Omega_2 &\le 1. \end{aligned} \tag{4}$$

The magnitude and angle of the continuous open-loop transfer function are

$$\begin{aligned} |X_s| &= M_1 \frac{\sqrt{M_2^2 + w^2}^3}{w\sqrt{M_4^2 w^2 + (M_5 - M_3 w^2)^2}} \\ \angle X_s &= -\pi + 3Arctan\Big(\frac{w}{M_2}\Big) - Arctan\Big(\frac{M_4 w}{M_5 - M_3 w^2}\Big). \end{aligned}$$

For the sake of system simplicity, the phase margin and magnitude should be independent of the average RTT. We thus consider the case of $w = M_2$ which represents a case of a break point frequency. We then investigate the relation between Ω_1 and Ω_2 so as that the crossover frequency is the same as the break point frequency, in other words, $(|X_s(w = M_2)| = 1)$. After calculation, we obtain the following condition

$$(1 - \Omega_2)^2 + \Omega_1^2 = \frac{1}{8}. \tag{5}$$

A specific case is when,

$$\phi = \frac{5}{4}, \psi = \frac{3}{4}.$$

REFERENCES

[1] N. Brownlee and K.C. Claffy, *"Understanding Internet Traffic Streams: Dragonflies and Tortoises"* , IEEE Comm. Mag., vol. 40, pp. 110-117, Oct. 2002.

[2] K. Thompson, G.J. Miller and R. Wilder, *"Wide-Area Internet Traffic Patterns and Characteristics"* , IEEE Network, vol. 11, pp. 10-23, Nov./Dec. 1997.

[3] G. Montenegro, S. Dawkins, M. Kojo, V. Magret and N. Vaidya, *"Long Thin Networks"* , Network Working Group, RFC 2757, Jan. 2000.

[4] L. Wood, G. Pavlou and B. Evans, *"Effects on TCP of Routing Strategies in Satellite Constellations"* , IEEE Comm. Mag., vol. 39, pp. 172-181, Mar. 2001.

[5] C. Partridge and T.J. Shepard, *"TCP/IP Performance over Satellite Links"* , IEEE Network. vol. 11, pp. 44-49, Sep./Oct. 1997.

[6] M. Allman, S. Floyd and C. Partridge, *"Increasing TCP's Initial Window"*, Network Working Group, RFC 2414, 1998.

[7] I.F. Akyildiz, G. Morabito and S. Palazzo, *"TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks"* , IEEE/ACM Trans. Networking, vol. 9, no. 3, pp. 307-321, June 2001.

[8] V. Padmanabhan and R. Katz, *"TCP fast start: A technique for speeding up web transfers"* , in Proc. IEEE GLOBECOM, Sydney, Australia, Nov. 1998, pp. 41-46.

[9] T. Henderson and R. Katz, *"Transport Protocols for Internet-Compatible Satellite Networks"* , IEEE J. Select. Areas Commun., vol. 17, no. 2, pp. 326-343, Feb. 1999.

[10] H. Balakrishnan, V.N. Padmanabhan and R. Katz, *"A comparison of mechanisms for improving TCP performance over wireless links"* , IEEE/ACM Trans. Networking, vol. 5, pp. 756-769, Dec. 1997.

[11] A. Bakre and B.R. Badrinath, *"I-TCP: Indirect TCP for mobile hosts"* , in Proc. 15th Int. Conf. Distributed Computing Systems (ICDCS), May 1995, pp. 136-143.

[12] S. Athuraliya, V.H. Li, S.H. Low and Q. Yin, *"REM: Active Queue Management"* , IEEE Network, vol. 15, pp. 48-53, May-June 2001.

[13] S. Floyd and V. Jacobson, *"Random Early Detection Gateways for Congestion Avoidance"* , IEEE/ACM Trans. on Networking, vol. 1, no. 4, pp. 397-413, Aug. 1993.

[14] K.K. Ramakrishnan and S. Floyd, *"Proposal to Add Explicit Congestion Notification (ecn) to IP"* , Network Working Group, RFC 2481, Jan. 1999.

[15] S.H. Low, F. Paganini, J. Wang, S. Adlakha and J.C. Doyle, *"Dynamics of TCP/AQM and a Scalable Control"* , in Proc. of INFOCOM, New York, June 2002.

[16] D. Katabi, M. Handley and C. Rohrs, *"Congestion Control for High Bandwidth-Delay Product Networks"* , in Proc. of SIGCOMM, Pittsburgh, PA, Aug. 2002.

[17] L. Kalampoukas, A. Varma and K.K. Ramakrishnan, *"Explicit Window Adaptation: A Method to Enhance TCP Performance"* , IEEE/ACM Trans. on Networking, vol. 10, no. 3, pp. 338-350, June 2002.

[18] J. Aweya, M. Ouellette, D.Y. Montuno and Z. Yao, *"WINTRAC: A TCP Window Adjustment Scheme for Bandwidth Management"* , Performance Eval., vol. 46, no. 1, pp. 1-44, Sept. 2001.

[19] T. Taleb, N. Kato and Y. Nemoto, *"On Improving The Efficiency and Fairness of TCP over Broadband Satellite Networks"*, in Proc. IEEE VTC, Orlando, Fl., Oct. 2003.

[20] T. Taleb, N. Kato and Y. Nemoto, *"An Explicit and Fair Window Adjustment Method to Enhance TCP Efficiency and Fairness over Multi-Hops Satellite Networks"*, IEEE J. Select. Areas Commun. vol. 22, no. 2, pp. 371-387, Feb. 2004.

[21] P. Fraise, B. Coulomb, B. Monteuuis and J.L. Soula, *"SkyBridge LEO Satellites: Optimized for Broadband Communications in the 21st century"*, in Proc. 2000 IEEE Aerospace Conf., vol. 11, Mar. 2000, pp. 1176-1183.

[22] Network Simulator - ns (version 2) [Online]. Available: http://www.isi.edu/nsnam/ns/

[23] S. Floyd and T. Henderson, *"The NewReno Modifications to TCP's Fast Recovery Algorithm"* Network Working Group, RFC 2582, Apr. 1999.

[24] R. Goyal and R. Jain, *"Buffer management and rate guarantees for TCP over satellite-ATM networks"* Int. J. Satell. Commun., vol. 19, pp. 111-139, 2001.

[25] J. Kulik, R. Coulter, D. Rockwell and C. Partridge, *"Paced TCP for High Delay-Bandwidth Networks"*, in Proc. of IEEE GLOBECOM, Dec. 1999.