An Explicit and Fair Window Adjustment Method to Enhance TCP Efficiency and Fairness Over Multihops Satellite Networks

Tarik Taleb, Nei Kato, *Member, IEEE*, and Yoshiaki Nemoto, *Member, IEEE*

***Abstract*—Transmission control protocol (TCP) is the most widely used transport protocol in today's Internet. Despite the fact that several mechanisms have been presented in recent literature to improve TCP, there remain some vexing attributes that impair TCPs performance. This paper addresses the issue of the efficiency and fairness of TCP in multihops satellite constellations. It mainly focuses on the effect of the change in flows count on TCP behavior. In case of a handover occurrence, a TCP sender may be forced to be sharing a new set of satellites with other users resulting in a change of flows count.**

This paper argues that the TCP rate of each flow should be dynamically 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 scheme matches the aggregate window size of all active TCP flows to the network pipe. At the same time, it provides all the active connections with feedbacks proportional to their round-trip time values so that the system converges to optimal efficiency and fairness. Feedbacks are signaled to TCP sources through the receiver's advertised window field in the TCP header of acknowledgments. Senders should accordingly regulate their sending rates. The proposed scheme is referred to as *explicit and fair window adjustment* (XFWA).

Extensive simulation results show that the XFWA scheme substantially improves the system fairness, reduces the number of packet drops, and makes better utilization of the bottleneck link.

***Index Terms*—Fairness, receiver's advertised window (RWND) adjustment, satellite networks, transmission control protocol (TCP), congestion-control.**

I. Introduction

TRANSMISSION control protocol (TCP) has been constantly in use in the Internet for over a decade. Nearly all applications that require reliable transmission use TCP as their transport protocol. Modern implementations of TCP consist of four mechanisms that were only recently fully documented as Internet standards: slow start, congestion avoidance, fast retransmit, and fast recovery. For a more comprehensive overview of these four algorithms, the interested reader is directed to [1].

Future broadband satellite networks intend to provide a wide variety of high and medium bit-rate data services to the end user and are capable of supporting flexible and scalable network configurations. Considering the fact that more than half of the world lacks a wired network infrastructure, and in order to achieve global coverage, satellite networks constitute an important alternative to current terrestrial networks. The design and development of satellite networks have been, thus, the subject of extensive research in recent literature [2], [3]. Given the dominance of TCP in today's Internet traffic [4], [5], it has become evident that TCP will be also the transport protocol of a significant part of the traffic of satellite networks.

TCP usually results in drastically unfair bandwidth allocations when multiple connections share a bottleneck link [13], [14]. 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 [4]. In case of multihops satellite constellations, since TCPs throughput is inversely proportional to the RTT, the unfairness issue becomes more substantial. When a group of users, with considerably different RTTs compete for the same bottleneck capacity, TCPs undesirable bias against long RTT flows becomes more significant, as is reported in [15].

Additionally, satellite networks are well characterized by frequent handover occurrences [16]. A handover occurrence may force a TCP sender to suddenly be sharing a new set of satellites with other users and resulting in an abrupt change in both the flows count and the connection's RTT. Fig. 1 illustrates a simple handover occurrence scenario, where a TCP connection is transmitting data over a link composed of three satellites. In the beginning, the TCP connection was utilizing the link bandwidth with no other flow. However, after handover occurrence, a group of new flows changes their path and starts sharing the same link with the TCP connection resulting in an increase of flows count. Let N_b and N_a denote the number of flows over a given link before and after handover occurrence, respectively. Assuming that TCP connections using the same link keep on transmitting data without any adjustment in their sending rates, two cases can be envisioned.

- $N_a < N_b$: The TCP sender will not make sufficient usage of the available bandwidth resulting in a waste of bandwidth and low link utilization.
- $N_a > N_b$: The TCP sender will be overloading the link with packets causing excessive queueing delays, large number of packet losses, and throughput degradation.

In order to maintain a high utilization of the bottleneck without overloading the network, to guarantee a fair share of the link bandwidth among all competing flows and, hence, to avoid high packet loss rate, the TCP sending rate of each

Manuscript received December 15, 2002; revised July 1, 2003 and September 20, 2003.

The authors are with the Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan (e-mail: taleb@nemoto.ecei.tohoku.ac.jp; kato@nemoto.ecei.tohoku.ac.jp; nemoto@nemoto.ecei.tohoku.ac.jp).

Digital Object Identifier 10.1109/JSAC.2003.819989

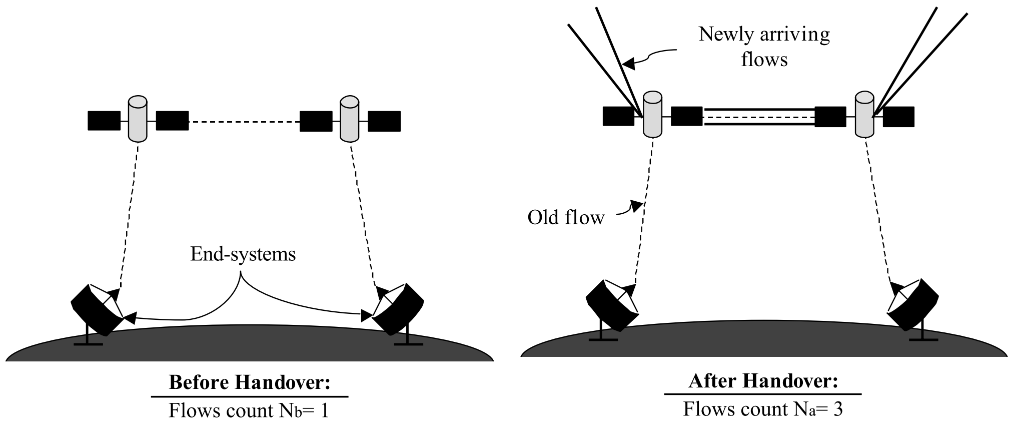

Fig. 1. Handover occurrence example.

flow should be constantly and fairly adjusted to the available bandwidth whenever traffic dynamics change.

The proposed scheme argues an explicit and fair control of all the TCP active connections' window sizes as a function of the effective bandwidth-delay product of the network. To control network utilization, the scheme matches the sum of the window sizes of all active TCP connections sharing the bottleneck link to the effective network bandwidth-delay product. Fairness, on the other hand, is achieved by assigning each connection a weight proportional to its round-trip time. Knowledge of the RTT estimates is usually not available at network elements in terrestrial networks. However, in case of multihops satellite constellations, certain features can help to compute an approximate value of the RTT as will be explained later. Once the desired window size is computed, network elements will signal the window update to the TCP source by modifying the receiver's advertised window (RWND) field carried by TCP ACKs. This operation can be accomplished without changing the protocol, and, as a result, requires no modification to the TCP implementations in the end systems. Moreover, modification of the RWND by intermediate hops in the network helps to communicate to the sender not only the flow control needs of the receiver but the flow control needs of these elements as well. The proposed scheme is referred to as *explicit and fair window adjustment* (XFWA).

Extensive simulation results demonstrate that the proposed scheme achieves high link utilization, provides low packet loss rates, reduces queue size, and maintains good fairness among the competing flows.

The rest of this paper is organized as follows. Section II discusses recent research work related to TCP/Internet protocol (IP) performance over satellite networks and TCP window adjustment areas. A detailed introduction of XFWA is given in Section III. Section IV describes the simulation setup. Simulation results are presented in Section V. Some implementation issues are discussed in Section VI and concluding remarks are presented in Section VII.

II. Related Work

As originally specified, TCP did not perform well over satellite networks for a number of reasons related to the protocol syntax and semantics [6]. To improve throughput performance of the TCP protocol in such environments, several proposals have been suggested in recent literature. To cope with slow start algorithm issues, the congestion window is initially set to a value larger than one packet size but lower than four packets size [7]. Other researchers have investigated the potential of a technique called TCP spoofing [8], [9]. 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 [10]. In TCP/shared passive network discovery (SPAND) [11], network congestion information is cached at a gateway and shared among many colocated 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. Reference [12] discusses the usage of low-priority dummy segments to probe the availability of network resources without carrying any new information to the sender. All in all, many researchers have investigated the efficiency and throughput improvement of TCP in satellite networks. However, most proposed solutions have considered only efficiency issues, mainly problems related to slow-start phase, and TCP behavior under many competing flows in satellite networks has not been sufficiently explored.

Current TCP implementations do not communicate directly with the network elements for explicit signaling of congestion control. TCP sources infer the congestion state of the 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 queueing, and intelligent packet-discard policies such as random early marking (REM) [17] and random early discard (RED) [18] combined with explicit congestion notification (ECN) [19]. These policies require a packet loss to signal the network congestion to TCP sources early. However, in case of large delay links (e.g., satellite links), these policies become inefficient and may still cause timeouts forcing TCP senders to invoke the slow-start phase. 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.* [20] have shown through mathematical analysis the inefficiency of these active queue management (AQM) schemes in environments with high bandwidth delay product such as satellite networks.

AQM limitations can be mitigated by adding 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. TCP-Vegas [21] attempts to compute optimal setting of the window size based on an estimate of the bandwidth delay product for each TCP connection. As knowledge of the RTT and the bandwidth delay product of the network is not usually available at network elements, TCP-Vegas requires extensive modifications to current TCP implementations in end-systems.

Katabi *et al.* [22] 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) [23] and window tracking and computation (WINTRAC) [24] suggest an explicit congestion control scheme of the window size of TCP connections as a function of the free buffer value similar in spirit to the idea of Choudhury *et al.* [25]. 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 toward connections with high variance in their RTTs distribution.

III. Explicit and Fair Window Adjustment (XFWA)

This section gives a detailed description of the proposed scheme XFWA. First, is an outline of the core ideas behind the scheme and its requirements.

A. *Per-Flow State Table*

The scheme requires maintaining per-connection state at the routers. Each router maintains a table of active flows with their flow ID, an estimate of their RTTs and their last packet transmission time. The flow ID is defined from source address, destination address, source port, destination port, and protocol. In case of Internet protocol version 6 (IPv6), the flow ID can be deduced simply from the flow label. At a given hop, prior knowledge of each flow's RTT can be handled by a simple monitoring of the backward and forward traffic of each flow as depicted in Fig. 2. Hops count of each flow can 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 queueing delays have minimal contribution in the one-way propagation delay and the value of the ISL delay remains constant in multihops satellite networks [27]–[29], the estimated value of the connection RTT can be expressed as

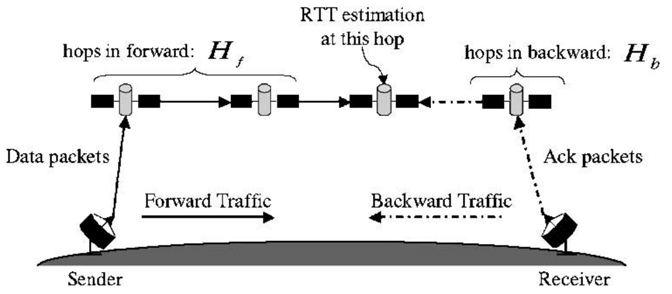

Fig. 2. Monitoring of the backward and forward traffic: Estimate of the RTT.

$$\mathrm{RTT} = 2 \cdot (H_b + H_f + 2) \cdot \mathrm{ISL}_{\mathrm{delay}} \tag{1}$$

where $\mathrm{ISL}_{\mathrm{delay}}$ denotes the ISL delay.

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. Unless stated otherwise, 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 $\mathrm{RTT}_{\mathrm{avg}}$ throughout this paper. Flows are counted periodically every $\mathrm{RTT}_{\mathrm{avg}}$ time and routers should create a new entry in the table whenever a new flow is detected. Note that estimating parameters over intervals longer than $\mathrm{RTT}_{\mathrm{avg}}$ will tend to ignore short flows leading to sluggish response, while estimating parameters over shorter intervals will lead to erroneous estimates, which ultimately affect the effectiveness of the proposed scheme [30], [31].

Due to the fundamental dynamics of TCP, there is a significant bias toward shorter connections when considering the individual throughput of connections with different RTTs. Being interested in quantifying this discrimination, several studies have shown that the average throughput of a TCP connection is inversely proportional to RTT^{β} [14], [26], where RTT is the connection's round-trip time and $1 \leq \beta < 2$. Assuming no queueing delays, the average throughput can be expected to be inversely proportional to RTT^2. The proposed scheme takes advantage of this attribute to make the system converge to max–min fairness.

B. *Feedback Computation Method*

When multiple TCP connections share a bottleneck link, XFWA matches the aggregate window size of all active TCP flows to the network pipe while at the same time providing all the connections with feedbacks proportional to their RTT values. This feedback computation causes the system to converge to optimal efficiency and fairness as will be shown through extensive simulation results. The feedback value of the ith TCP connection is computed as

$$\text{feedback}_i = \frac{\text{RTT}_i^{\alpha}}{\sum_{j=1}^{N} \text{RTT}_j^{\alpha}} (Bw \cdot \text{RTT}_{\text{avg}} + Q_{\text{size}}) \tag{2}$$

where N, Q_{size}, and Bw are the total number of TCP flows traversing the router, the router's queue size, and the link bandwidth, respectively. RTT_{avg} is the average RTT and α is a constant parameter, whose influence on the system efficiency and fairness will be discussed later in the simulation results. In the rest of this paper, α is referred to as the *skew factor* of the system. Setting α to values larger than one aggravates TCPs undesirable bias toward short RTT flows while setting α to values smaller than one yields a bias in favor of long RTT flows.

One of the most interesting attributes of this feedback computation method is that it allows the scheme to automatically adapt to the number of active TCP flows, the buffer size, and the bandwidth delay product of the network. Another benefit is that it controls the efficiency of the system by matching the aggregate traffic rate to the link capacity and total buffer size. This attribute eventually helps to adjust the protocol's aggressiveness and to prevent persistent queues from forming. To achieve min–max fairness, the first term of (2) reallocates bandwidth between individual flows in proportion with their RTTs. Appendix I shows that the implementation of XFWA scheme is fairly simple and requires only a little computational load.

C. Feedback Signaling Method

The window feedback is computed every RTT_{avg} time and is written in the RWND field carried by the TCP header of acknowledgment packets similar in spirit to the EWA [23] and WINTRAC [24] approaches. RWND adjustment requires the ability of the router to separate ACK packets from data packets. This ACK packet identification can be easily carried out by examining the ACK bit in the TCP packet header.[1] On the other hand, RWND adjustment neither requires modifications to the protocol implementations in the end system, nor does it modify the TCP protocol itself.

The RWND value can only be downgraded. If the original value of the RWND, which is set by the TCP receiver, exceeds the feedback value computed by a downstream node, the RWND 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 RWND field. Ultimately, the RWND field will contain the optimal feedback from the bottleneck along the path. Values of RWND smaller than the optimal value decrease the throughput and result in link underutilization, whereas larger values of RWND contribute only to increased queueing delays, multiple drops and, hence, throughput degradation. 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.

[1]ACK packets always have their ACK bit set by the TCP receiver.

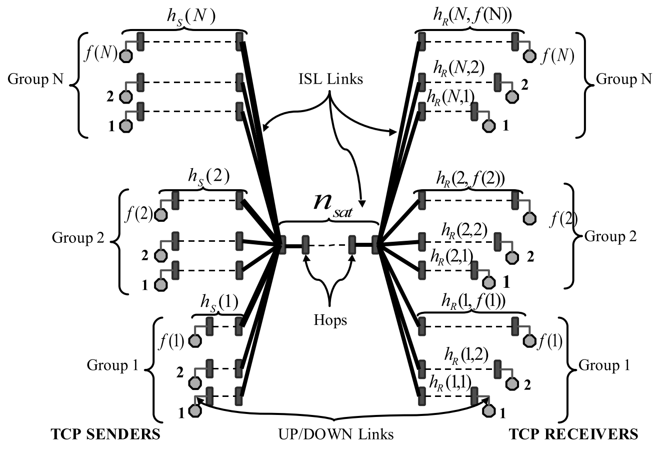

Fig. 3. Simulation topology.

D. Coexistence of TCP Flows With Nonresponsive Traffic

So far, we have considered only TCP flows competing for the capacity of a given bottleneck link. We can, however, extend our studies to more general scenarios, where nonresponsive traffic, such as user datagram protocol (UDP), coexists with TCP flows. In such a case, the incoming traffic can be mapped into different traffic classes based on IP addresses, the type-of-service (ToS) field, and port numbers. Each traffic class will be managed by a per-class queue, which functions according to a specific queueing discipline. Application of the proposed scheme is then relevant to only the TCP queueing class.

IV. Simulation Environment

Having described the details of the XFWA scheme, we now direct our focus to evaluating its performance. The performance evaluation relies on computer simulation, using network simulator (NS) [32]. Therefore, particular attention is paid to the design of an accurate and realistic simulation setup, which is described in this section, justifying the choices made along the way. Unless otherwise noted, the parameters specified below are those used in all the experiments in the paper.

To illustrate the issues at hand, a satellite network is modeled as a one network bottleneck shared by various connections. Fig. 3 shows the example network configuration used in this paper. The abstract configuration consists of three parts.

- *Bottleneck link*: The link is composed of a given number of satellites. The total number of these hops is denoted as n_{sat}.
- *TCP senders*: This side consists of N groups of flows. Each group G_i contains $f(i)$ flows traversing the same number of hops $h_s(i)$ before entering the bottleneck link.[2] For the sake of simplicity, all the $f(i)$ flows are assumed to traverse different set of satellites before entering the bottleneck as depicted in Fig. 3. This assumption has no effect on the overall performance of the proposed scheme.
- *TCP receivers*: Similarly to TCP senders, receivers are also grouped in N groups. The ith group is the set of receivers connected to senders that belong to group G_i at

[2]The total number of sources connected to the bottleneck is $f_{\text{total}} = \sum_{i=1}^{N} f(i)$.

sources side. The function $h_r(i,j)$ $(\forall i \in [1,N], \forall j \in [1,f(i)])$[3] points to the number of hops traversed by the jth flow in the ith group after exiting the bottleneck link.

In the remainder of this paper, the following notation is used to refer to the underlying topology:

$$\mathrm{Topology}(n_{\mathrm{sat}}, N, f(i), h_s(i), h_r(i,j))$$

where $h_s(i)$ and $f(i)$ are functions with definition interval $[1,N]$, and $h_r(i,j)$ is a function with definition interval $[1,N] \times [1,f(i)]$.

As for the links capacity, a number of test scenarios were created by setting the ISL bandwidth Bw to different typical link speeds: 1.5 (e.g., T1), 10 (e.g., T2), 45 (e.g., T3), and 155 Mb/s (e.g., OC3). Unless otherwise stated, the ISL delay $\mathrm{ISL}_{\mathrm{delay}}$ is set to 20 ms and all uplinks and downlinks are given a capacity equal to 10 Mb/s. These parameters are chosen with no specific purpose in mind and do not change any of the fundamental observations about the simulation results. All links are presumed to be error-free throughout this paper. 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.

In all simulations, TCP sources implement the TCP NewReno version [34]. TCP NewReno achieves faster recovery and has the potential of significantly improving TCPs performance in case of bursty losses. While it has been observed that many Internet connections are of short durations, the behavior of the proposal is best understood by considering persistent sources, which could be thought of as modeling long file transfers. TCP connections are, therefore, modeled as greedy long-lived FTP flows. These long-lived FTP flows can serve to model real video streams as well, where efficiency and fairness issues matter as the number of video-data receivers increases. In order to investigate the dynamics of the proposed scheme with web-like traffic, a scenario where a mix of greedy and nonpersistent flows competes for the bandwidth of the bottleneck link, is also considered. The data packet size is fixed to 1 kB. 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 [33]. Throughput and queue size measurements are performed in intervals of 100 ms. Due mostly to its simplicity and its wide usage in today's switches and routers, all routers use drop-tail as their packet-discarding policy. Simulations were all run for 20 s, a duration long enough to ensure that the system has reached a consistent behavior.

The conducted simulations cover number of flows in [1–512] and the maximum number of connections varies according to the type access link. For each access link, a maximum number of flows MAX is fixed so that a minimum value of link fair share can be guaranteed for all competing connections (Table I).

In order to illustrate the effect of the change in flows count on TCP behavior and examine how the XFWA scheme adapts to sudden increases or decreases in traffic demands, the four scenarios presented in Fig. 4 are considered.

- Scenario A: In this scenario, the flows count remains constant throughout the simulation and is equal to MAX. All connections are activated at the beginning of the simulation. Their starting time is uniformly distributed from 0 to 5 ms to avoid bursty losses at the simulation launch time. All connections remain open for 20 s, the simulation running time.
- Scenario B: This scenario depicts a periodic increase of flows. The simulation is launched with a quarter of the maximum number of flows $(1, (\mathrm{MAX}/4))$. At time $t = 5$ s, we start another quarter of flows $((\mathrm{MAX}/4) + 1, (\mathrm{MAX}/2))$ and let them stabilize. The same operation is repeated at time $t = 10$ s opening the third quarter of flows $((\mathrm{MAX}/2) + 1, (3 \cdot \mathrm{MAX}/4))$. At time $t = 15$ s, the last quarter of flows $((3 \cdot \mathrm{MAX}/4) + 1, \mathrm{MAX})$ are activated.
- Scenario C: This scenario considers the case of a periodic decrease of connections. The total number of connections simultaneously open at time $t = 0$ and remain active for 5 s. At time $t = 5$ s, a quarter of the flows $((3 \cdot \mathrm{MAX})/(4) + 1, \mathrm{MAX})$ close. The remaining connections stay active for another 5 s, when another quarter of the connections $((\mathrm{MAX})/2) + 1, (3 \cdot \mathrm{MAX}/(4)$ close. At time $t = 15$ s, another quarter of flows $((\mathrm{MAX})/(4) + 1, (\mathrm{MAX})/(2))$ close. The remaining connections stay active until the end of the simulation at time $t = 20$ s.
- Scenario D: This scenario can be thought of as a combination of scenarios B and C as it considers the case of both a periodic increase and decrease of flows. In this scenario, the simulation starts with half of the flows $(1, (\mathrm{MAX})/(2))$ that are left active for 5 s, when the 2nd half of flows $((\mathrm{MAX})/(2) + 1, \mathrm{MAX})$ are launched. At time $t = 10$ s, a quarter of the flows $((\mathrm{MAX})/(4) + 1, (\mathrm{MAX}/2))$ close. After 5 s, another quarter of flows $((\mathrm{MAX})/(2) + 1, (3 \cdot \mathrm{MAX})/4)$ is deactivated. The remaining connections are left active until the end of the simulation.

TABLE I
MAXIMUM FLOWS COUNT

Link Type	Maximum number of flows (MAX)
T1	8
T2	32
T3	128
OC3	512

Due to handover occurrence in a multihops satellite network, a TCP connection may either alter its path and compete for bandwidth with a different group of connections, or just keep its path but be forced to be sharing the same link with newly coming connections. Both cases will result in an abrupt change in the number of flows and this justifies the choice of the stair-step graph in the four considered scenarios (Fig. 4).

After this detailed description of the simulation setup, we now present the performance measures used to evaluate the performance of the proposed scheme. In addition to TCP sequence

[3] $[1,M] = \{k \mid k = 1,2,\ldots,M\}$.

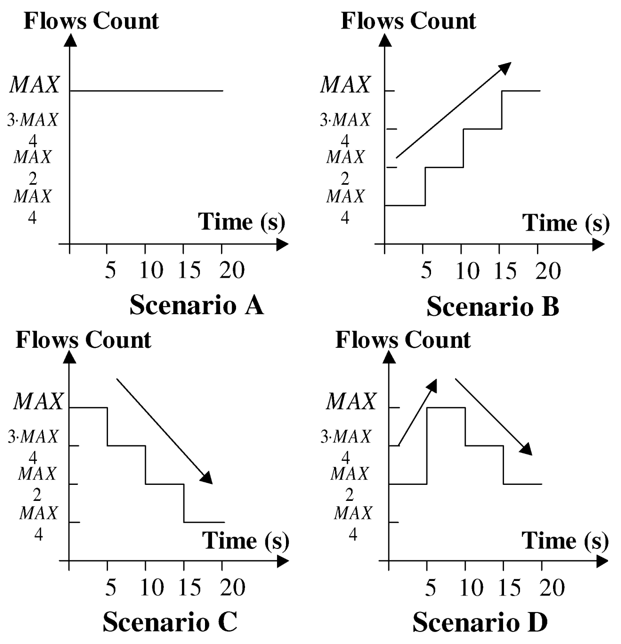

Fig. 4. Flows count change scenarios.

number and individual flow throughput metrics that give good insight into the behavior of individual flows, the following measures will be used to capture the aggregate behavior of flows and the overall network performance.[4]

- Bottleneck link utilization: Ratio of the aggregate throughput to the bottleneck link capacity. This measure involves the aggregate traffic's behavior and indicates the efficiency of the protocol.
- Loss rate: Ratio of the dropped packets to the aggregate sent packets.
- Average queue size: The average queue occupancy at the bottleneck over the transmission time measured every 100 ms.
- Link average throughput: This measure indicates the number of bytes received in each 100 ms time interval.
- Fairness index: This measure is defined in [35]. The fairness index involves the relative throughput of flows sharing a link defined as

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

where x_i is the actual throughput of the ith flow and b_i is the equal share of the bottleneck link capacity. The fairness index of a system ranges from zero to one. Low values of the fairness index represent poor fairness among the competing flows. Depending on the application and the number of TCP senders, gaining higher fairness values is sometimes worthwhile even at the cost of reduced efficiency.

[4]The aggregate performance is sometimes more interesting as it is often more useful for network provisioning.

V. Performance Evaluation

In this section, the system performance when the XFWA scheme is used to control TCP windows is evaluated through extensive simulation experiments. In the performance evaluation, comparison is made between the dynamics of XFWA and those of standard TCP in both steady-state and dynamic environments considering the case of TCP flows with same RTTs and TCP flows with different RTTs. In addition to persistent TCP sources, the performance of the scheme is also investigated under ON–OFF sources, where the computed feedback is believed to be less effective.

A. Robustness to Dynamic Changes in Traffic Demands: Flows With Same RTTs

In this experiment, the used network configuration is topology (3, 1, MAX,[5] 0, 0). Note that all flows traverse the same number of hops (three satellites) resulting in each source having an RTT of 160 ms. The network topology depicted in Fig. 3 becomes a symmetric and simple network bottleneck shared by all connections.

Fig. 5 graphs the overall network performance in terms of link utilization, drops rate, and fairness for different access link types and different scenarios of change in flows count. Simulation results show no cases where XFWA performs worse than TCP. In fact the results show that the scheme exhibits better fairness, higher utilization of the bottleneck link, and smaller number of drops. Without the XFWA 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. This behavior causes the TCP sources to decrease their transmission rates and results in an overall performance degradation. However, in the case of the XFWA scheme, the system is always self-adaptive to the traffic load and the number of active connections: the aggregate window feedbacks of all active TCP flows remains bound by the bandwidth delay product of the network. This attribute allows TCP sources to achieve optimal performance by reducing traffic bursts and explains the better performance of the XFWA scheme in terms of lower loss rates and higher utilization of the link. Simulation results show also that better fairness values are achieved in case of XFWA than standard TCP and that is for all considered scenarios. Although the "surpass" in the fairness index is sometimes minimal, we shall not ignore that a difference in the order of 10^{-2} in fairness index can largely affect the fairness of the system, as has been indicated in several previous studies. Since all flows have the same propagation delay, the experienced unfairness is mainly due to packet losses. The loss rate and fairness index results inspire a certain correlation between the two measures.[6] It is generally observed that when the system suffers a higher packet loss rate, it tends to be more unfair. In case of scenario A, the loss rate of TCP is significant. This is likely because many of the connections, if not all of them, experienced losses at the same time. These synchronized losses cause the connections to simultaneously reduce their windows and ultimately re-

[5]Depends on the access link type.

[6]Loss rate diagrams look as if they are the upside-down versions of the corresponding fairness index diagrams and *vice versa*.

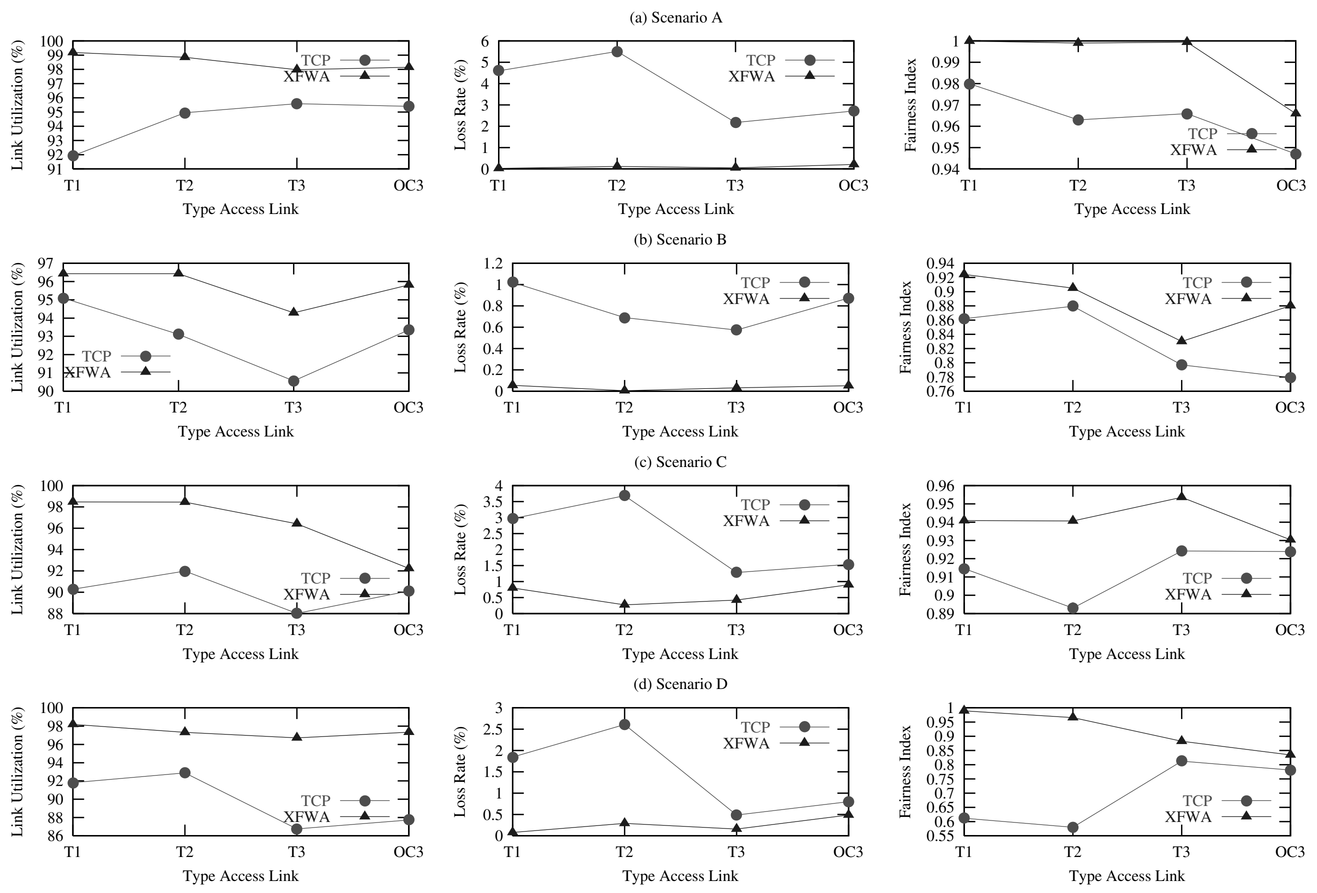

Fig. 5. Overall performance (flows with equal RTTs).

sult in relatively similar overall throughputs among all flows. This explains the higher values of fairness index of TCP experienced by the system in case of scenario A. In case of the other dynamic scenarios (B, C, D), the system becomes significantly unfair despite the relatively lower loss rates (in comparison to the experienced drop rate in scenario A). This is because in dynamic environments, already existing TCP connections keep on sending data with high rates preventing newly coming TCP connections from a fair bandwidth allocation. When some connections leave the network, competition for bandwidth among remaining connections becomes significant resulting in packet drops. As a consequence, connections that suffer lower packet loss tend to have a higher transmission rate and connections that suffer higher packet loss are knocked down to smaller transmission rates. The XFWA, however, always maintains higher fairness index values. The underlying reason for this performance consists in the XFWAs ability to rapidly adapt to changing network conditions. When new connections enter the network, the XFWA scheme quickly computes a smaller window feedback and when some connections exit the network, it feeds back TCP sources with larger window sizes.

In order to demonstrate how the XFWA scheme achieves better fairness, the growth of TCP sequence numbers for each simulated TCP connection is plotted. Since all flows have the same propagation delay, the results of scenario A are of little significance and do not serve to demonstrate the convergence dynamics of the XFWA algorithm. Focus is, thus, on discussing the results obtained in case of the other dynamic scenarios. Since the number of TCP sources is large in case of the access links $T2, T3$, and $OC3$, and for ease of illustration for the plots, only the segment number growth when the bottleneck link is $T1$ is plotted.

In Fig. 6(a), the second flow gets some of its packets dropped when the third and fourth connections enter the network at time $t = 5$ s. This causes the second flow to transmit data at a window size smaller than its fair share value. On the other hand, because of TCPs greediness, the first flow makes use of the available bandwidth and increases its window size to values higher than its fair share value. This explains the abrupt increase of segment number of the first flow. Observe also that at time $t = 10$ s and $t = 15$ s, the slope of the sequence number growth of the newly entering flows (fifth and eighth) exhibits some small oscillations most probably due to timeouts. However, with XFWA, it is observed that none of the simulated flows experienced a timeout, and, that the increase in the sequence number of all flows is parallel. This is because, unlike TCP, the XFWA scheme attempts to fairly divide the available bandwidth among the competing flows and 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 progres-

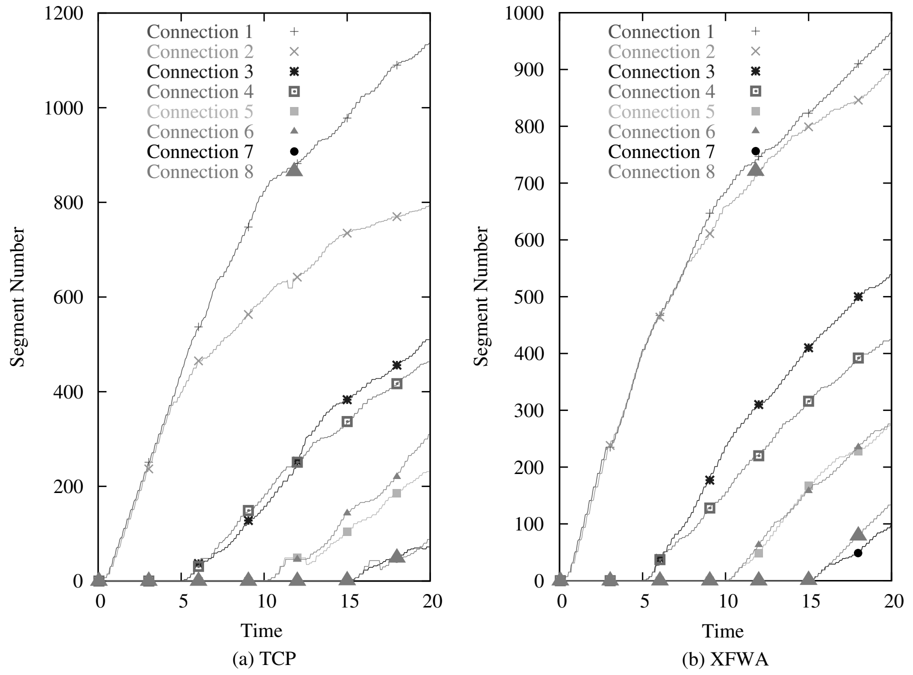

Fig. 6. Sequence number growth (flows with equal RTTs, access link T1, scenario B).

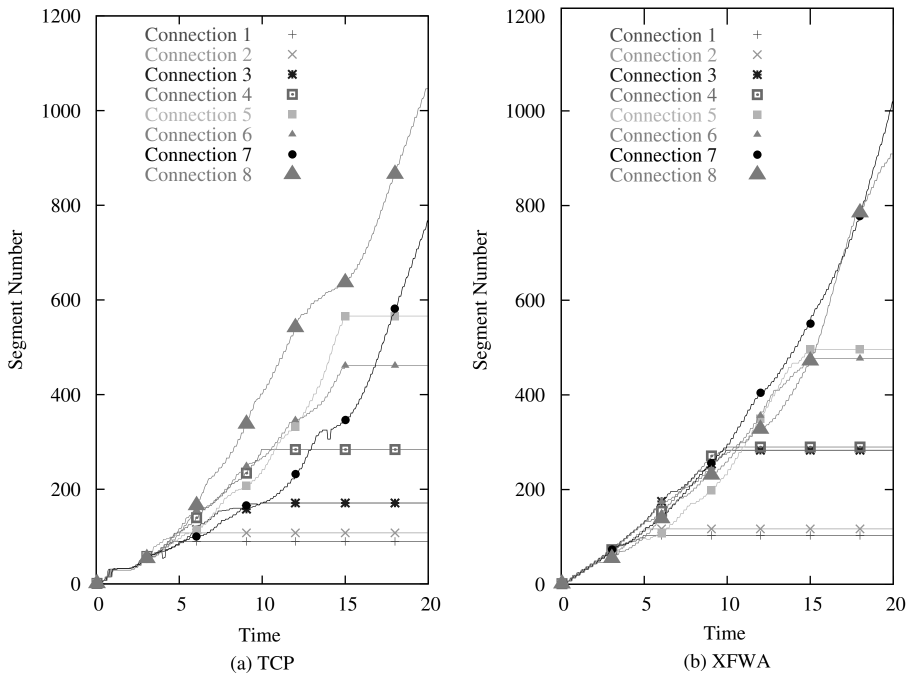

Fig. 7. Sequence number growth (flows with equal RTTs, access link T1, scenario C).

sion in window size for both newly coming and already existing flows. Figs. 7 and 8 illustrate the sequence number growth of the active TCP connections in case of scenario C and D, respectively. In both scenarios, the XFWA scheme significantly outperforms TCP: the TCP segment number of all TCP connections progress evenly when XFWA is used. In scenario C, when the number of active connections decreases at time $t = 5$ s, $t = 10$ s, and $t = 15$ s, XFWA helps the remaining active connections to detect and to rapidly recover the unused bandwidth while maintaining fairness. In scenario D, with XFWA, the slope of the sequence number lines decreases at time $t = 5$ s as four new flows enter the system, and increases at time $t = 10$ s and $t = 15$ s as a result of extra bandwidth becoming available for the remaining active connections. Observe that no timeout was recorded in case of XFWA, whereas quite a few flows suffered timeouts in both scenarios when the TCP windows are controlled by only TCP.

The XFWA outperforms standard TCP not only in terms of fairness, link utilization and packet drops, but also in minimizing the queue size. Due to paper length limitation, we present only the queue size variation of the bottleneck link in case of scenario D. This latter considers the case of both an

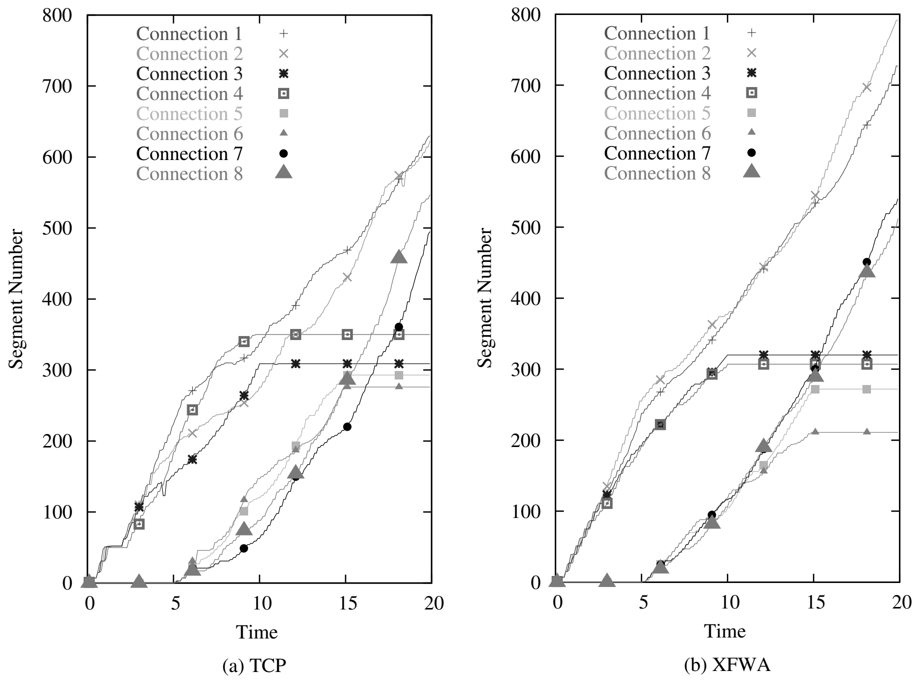

Fig. 8. Sequence number growth (flows with equal RTTs, access link T1, scenario D).

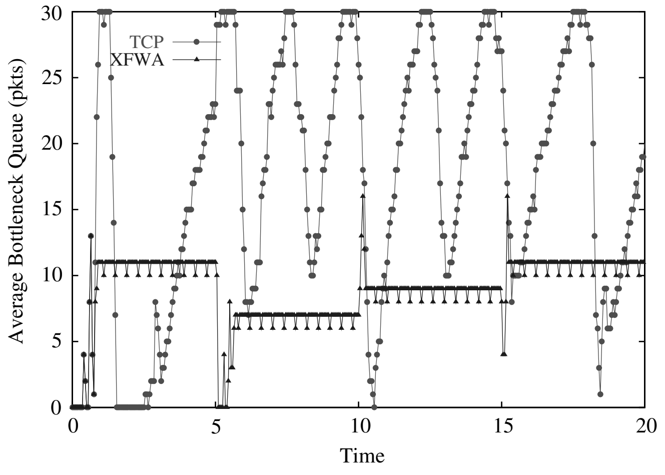

Fig. 9. Average bottleneck queue (flows with equal RTTs, access link T1, scenario D).

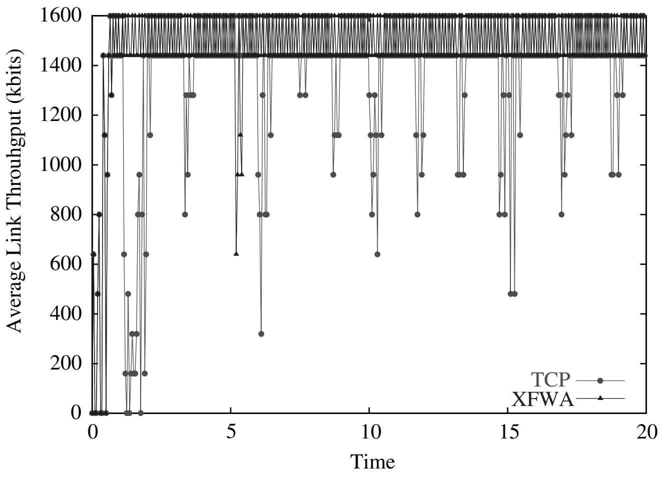

Fig. 10. Average bottleneck link throughput (flows with equal RTTs, access link T1, scenario D).

increase and decrease in the flows count and its results are highly representative of the general behavior of scenarios B and C. The queue occupancy of the bottleneck link, measured in intervals of 100 ms is presented in Fig. 9.

The figure indicates that without XFWA, 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, the XFWA scheme is effective at protecting the buffer from overflows. It is also self-adaptive to the change in traffic dynamics and the number of active connections. With XFWA, the buffer underflows that are mostly due to the change in traffic demands are brief and do not significantly affect the bottleneck link utilization.

Fig. 10 shows the bottleneck link throughput averaged over intervals of 100 ms. Without XFWA, the throughput fluctuates irregularly. The throughput loss is mainly due to the synchronization of the packet losses of the connections and their simultaneous recovery. In contrast, XFWAs utilization is always near the total capacity of the link and exhibits very limited oscillations.[7] In conclusion, the XFWA scheme manages to control the buffer occupancy well and achieves stability and robustness when a change in traffic load occurs. The average link utilization, as graphed in Fig. 10, is perfect most of the time.

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

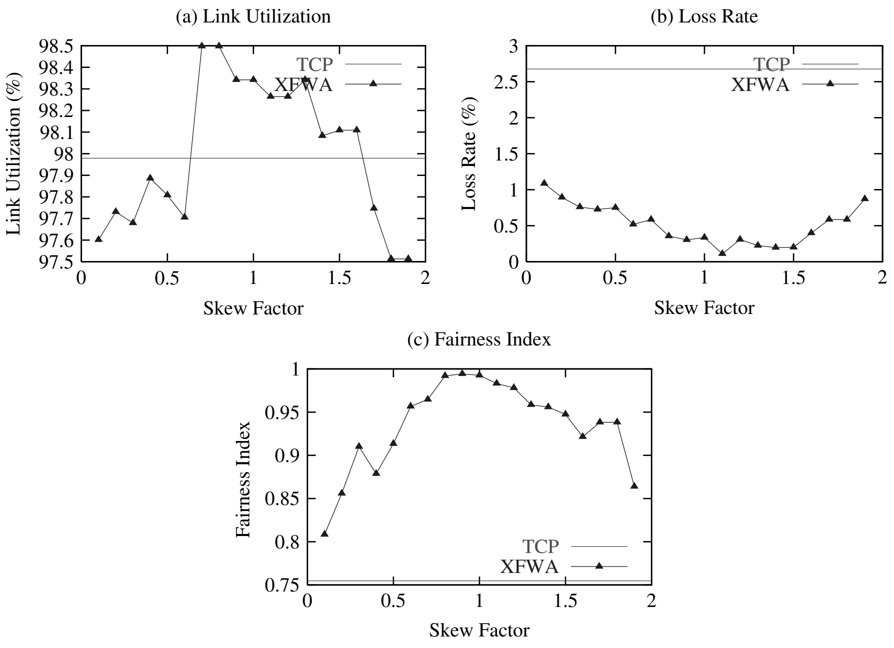

Fig. 11. Skew factor effect on the system performance (scenario A, access link T1).

B. Fairness: Flows With Different RTTs

The previous simulation results have demonstrated the robustness of XFWA to sudden increase or decrease in traffic demands. Henceforth, we examine the robustness of XFWA to variance in the flows RTT distribution. This section aims to show through simulation results that XFWA is significantly fairer than TCP and has no bias against long RTT flows. The used network configuration is $\mathrm{Topology}(3, 4, (\mathrm{MAX}/4), h_s(i), h_r(i,j))$. $h_s(i)$, and $h_r(i,j)$ are defined as follows:

$$h_s(i) = i - 1 \quad \forall i \in [1, 4],$$
$$h_r(i,j) = i - 1 \quad \forall j \in \left[\frac{(i-1)\mathrm{MAX}}{4}, \frac{i\mathrm{MAX}}{4}\right].$$

Considering the dynamic scenarios C, B, and D is useful in that it demonstrates the convergence dynamics of the XFWA scheme and captures its fairness. However, it is too complicated to obtain a comprehensive understanding of XFWA behavior in such scenarios. The main reason for this difficulty is that it is all but impossible to determine which flow should enter or leave the system. Because of the additive increase multiplicative decrease (AIMD) behavior of TCP, activating short RTT flows before long RTT connections tends to increase the link utilization, whereas starting long RTT flows earlier than short RTT connections lowers the link utilization. In order to take into account all the possibilities of flow entry/departure determination, many simulations should be set up. For ease of exposition, discussion is made only on the results of scenario A, where the number of active flows remains stable during the course of the simulation. For the overall performance of the system, the reader is referred to Appendix IV.

We first investigate the effect of the skew factor α on the system performance. Fig. 11 presents the overall network performance in terms of link utilization, drops rate, and fairness index for different values of α. The figure presents the system performance when the bottleneck link is T1. However, same experiments were repeated for other link speeds and identical results were obtained. The figure demonstrates that setting α to values larger than one worsens TCPs undesirable bias toward short RTT flows. This aggravation is manifested in the form of lower values of fairness index, higher loss rates, and degraded link utilization. Values of α larger than one, however, yield a bias in favor of long RTT flows. This bias causes the long RTT flows to send data with rates higher than their fair shares resulting in traffic bursts. On the other hand, short RTT flows will be allocated portions of bandwidth smaller than what it is required to achieve optimal performance. This behavior ultimately causes the system to perform poorly and results in a significant degradation of the link utilization and higher number of drops. Unless otherwise stated, the skew factor α is set to one in the remainder of this paper.

The sequence number growth of the TCP connections in the case of scenario A is plotted in Fig. 12. Unlike standard TCP, which is grossly unfair toward connections with higher RTTs, the XFWA scheme attempts to fairly divide the available bandwidth among all competing flows, while taking into account each flow's RTT. This has led to a fair progression in window size for all flows as indicated in Fig. 12(b), where the sequence number increase of all connections is parallel. This fair progression in window size can be deduced as well from Fig. 13, where all the flows could send nearly the same amount of packets in case of the XFWA scheme. 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 with the long RTT connections. The obtained results confirm that the XFWA estimate of the average RTT of the system operates correctly in environments with high variance in the RTT distribution.

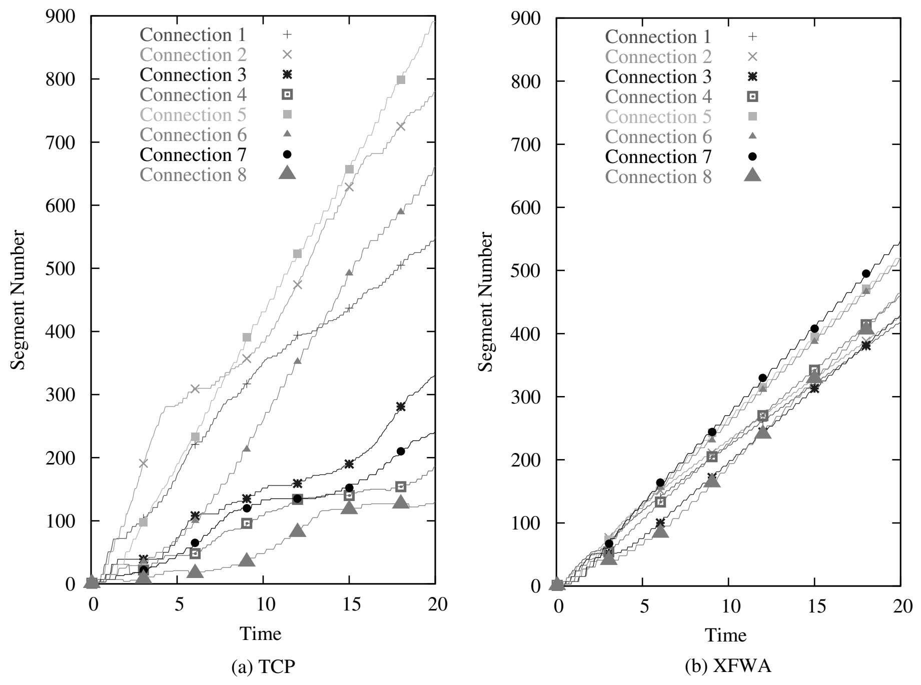

Fig. 12. Sequence number growth (flows with different RTTs, access link T1, scenario A).

C. Resiliency to Reasonable RTT Estimation Errors

Appendix II provides a simple mathematical analysis showing that the XFWA scheme maintains a fairly good performance even in the presence of reasonable RTT estimation errors. Henceforth, the resiliency of XFWA to RTT errors is examined through simulation results.

The considered network configuration is that of Section V-B.[8] As previously described, the RTT estimation is made based on the number of hops. Therefore, any RTT estimation error value[9] should be an even multiplication of $\mathrm{ISL}_{\mathrm{delay}}$. In this simulation, in order to study the overall performance of the system for different RTT error values, this fact is however ignored, and error ratios[10] are deliberately derived from a uniform distribution with zero mean and variance σ^2. In NS implementation, the extreme case is considered by setting the maximum and minimum value of the distribution to σ and $-\sigma$, respectively, $(\max\_ = \sigma, \min\_ = -\sigma)$.

Fig. 14 presents the overall network performance in terms of link utilization, drops rate, and fairness index for different values of σ. The figure confirms the results of the mathematical analysis developed in Appendix II. It demonstrates that for small values of σ, the XFWA scheme still maintains good performance: better link utilization and higher values of fairness index. However, large values of σ worsens the system performance. This aggravation is manifested in the form of lower values of fairness index and link utilization degradation.

[8]The same experiment was conducted considering the network topology of Section V-A and similar results were obtained.

[9]The difference between the erroneous and good estimates of RTT.

[10]Defined as the ratio of the difference of the erroneous and good estimates of RTT to the good estimate of RTT (see Appendix II).

D. Performance Evaluation With On–Off TCP Flows as Cross Traffic

Since a large number of flows in today's Internet are short web-like flows, the interaction and resulting impacts of such dynamic traffic on the XFWA scheme are discussed in the remainder of this section. It has been reported in [36] that web-like traffic tends to be self-similar in nature. In [37], it is shown that self-similar traffic can be modeled as several ON–OFF TCP sources whose ON–OFF periods are drawn from heavy-tailed distributions such as the Pareto distribution. In this experiment, a scenario where a mix of ten long-lived FTP flows and a number of nonpersistent flows compete for the bottleneck link bandwidth is considered. The considered network configuration is $\mathrm{Topology}(3, 1, 10 + \mathrm{Count}_{\mathrm{OnOff}}{}^{11}, 0, 0)$ and the used access link type is T2. The simulation starts with 1ten persistent connections at time $t = 0$ s. The persistent TCP flows remain open until the end of the simulation. At time $t = 5$ s, the ON–OFF TCP flows are activated and remain open for a duration of 10 s. The ON–OFF periods of the nonpersistent connections are derived from Pareto distributions with the mean ON period and the mean OFF period set to 160 ms, a value on the average equal to the flows RTT. This choice is deliberately made to prevent the ON–OFF TCP sources from entering the slow-start phase even after periods of idleness. This will, thus, help to illustrate the resiliency of the XFWA even in the case of significant amount of burstiness in the network.

Fig. 15 shows the bottleneck utilization and drops rate for different number of ON–OFF flows count. The results demonstrate the good performance of the XFWA scheme even in environments with bursty traffic. The scheme maintains higher utilization of the bottleneck link and significantly reduces the number

[11]ON–OFF flows count.

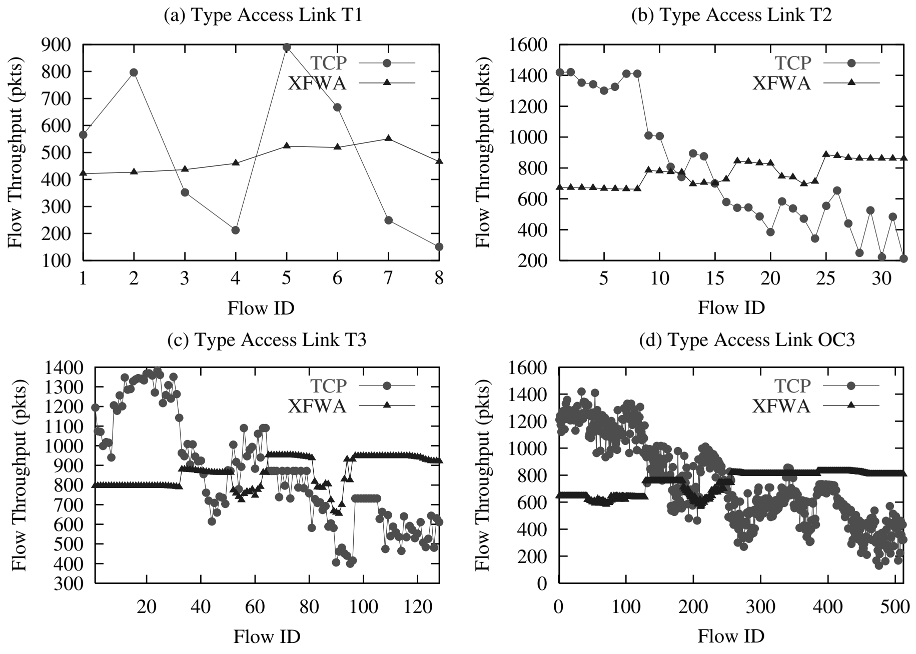

Fig. 13. Individual flow throughputs for different access links (flows with different RTTs, scenario A).

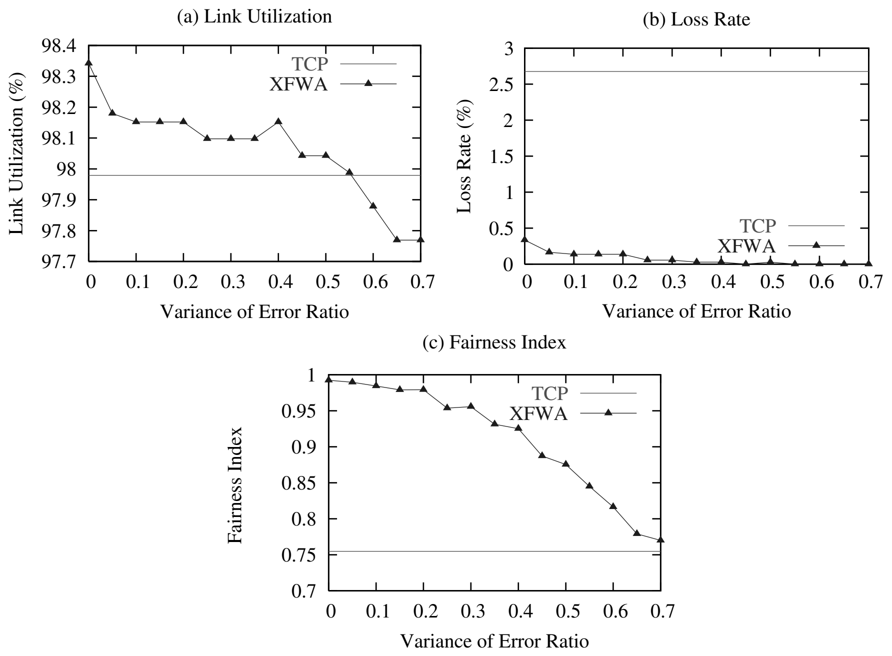

Fig. 14. System resiliency to RTT estimation errors (flows with different TTs, Scenario A, access link T1).

of drops even for higher number of ON–OFF flows. This is because unlike standard TCP, the XFWA algorithm attempts to bound the aggregate window sizes of all active TCP flows to the bandwidth delay product of the network and, thus, avoids overloading the bottleneck link with packets. The sequence number growths of the ten persistent TCP connections and some of the ON–OFF flows are plotted in Fig. 16. Since the number of ON–OFF TCP sources is large (case of 45 ON–OFF flows), the sequence numbers of only source 15 and every other fifth source thereafter are plotted. The figure confirms the ability of the XFWA scheme to achieve fairness even in the presence of traffic bursts. With the XFWA scheme, the progress of all persistent TCP connections remains fair during the running time of the simulation. Observe how the slope of the sequence number lines changes at time $t = 5$ s and $t = 15$ s as a result of entry and departure of ON–OFF flows, respectively. Note also that in case of only standard TCP, both ON–OFF and persistent flows exhibit great deviations, whereas XFWA helps all flows to fairly progress and to behave in an identical way. These results show that the XFWA scheme does not penalize the ON–OFF traffic for being idle and, thus, demonstrate the resiliency of the scheme to accommodate such dynamic traffic.

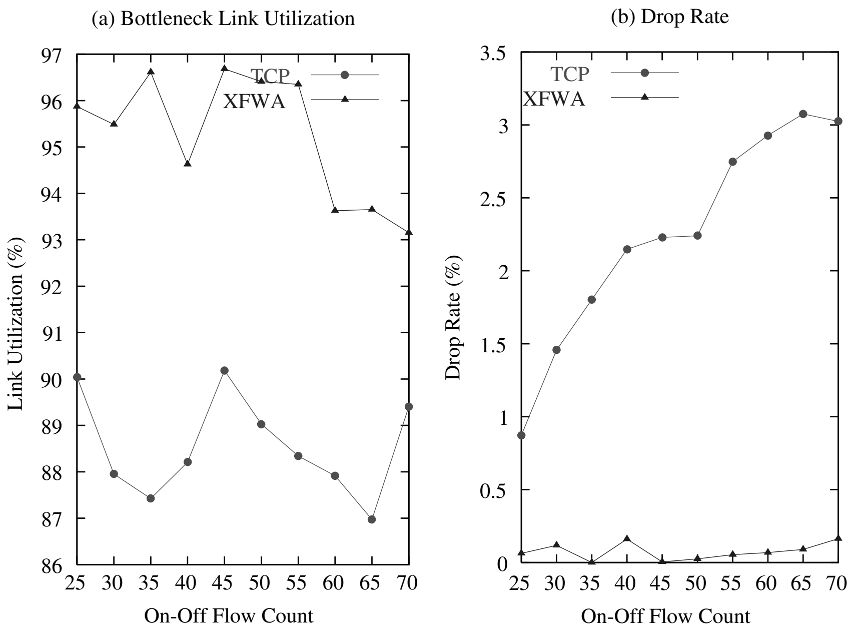

Fig. 15. Link utilization and loss rate in the presence of web-like traffic (persistent flows count 10, access link T2).

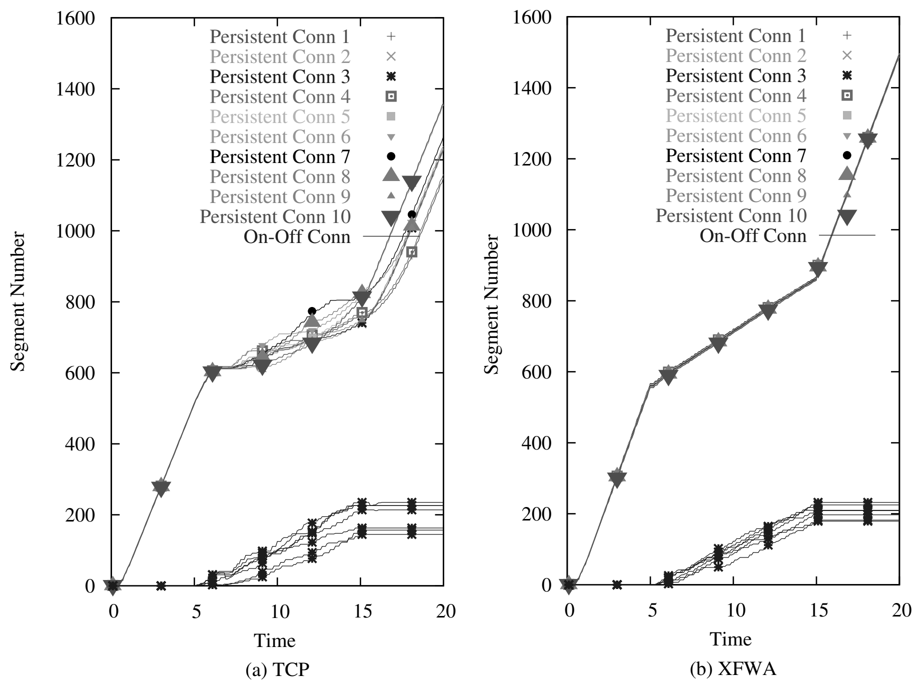

Fig. 16. Sequence number growth (persistent flows count 10, ON–OFF flows count 45, access link T2).

VI. Implementation Issues and Discussion

It should be emphasized that there are several implementation issues that must be resolved when applying the XFWA scheme to practice. For instance, an XFWA satellite should be capable of reading and modifying the TCP header in packets. This violates the IP-Security (IPSec) semantics according to which packets must be processed by only the end-systems; no third party is allowed to modify the payload. Despite such violations, a number of schemes requiring flow identification and using window adjustment have been proposed in literature [23], [24]. Another concern about the XFWA is that packets and their acknowledgements must follow the same path if estimate of flows RTT and TCPs advertised window adjustment are to be effective. This limitation can, however, be solved by careful design of routing approaches. Appendix III discusses the possible cases in which this limitation might affect the XFWAs performance and suggests some approaches to tackle it.

The XFWA could be costly in terms of other resources. For example, since an XFWA satellite should maintain state information for individual TCP flows, this will obviously incur more overhead at the system in terms of memory, few bytes per flow. However, the obtained performance gains (higher fairness, higher link utilizations, and lower packet loss rates) are worthwhile and can be used to advocate this additional cost. As for the feedback computation, it is fairly simple and requires only a few additions and a few multiplications every $\mathrm{RTT}_{\mathrm{avg}}$ interval of time, as explained in Appendix I.

In the conducted simulations, a simple network environment was considered. It was assumed that only one-bottleneck is traversed by many connections and that all flows are always making full use of their allocated window. It was also assumed that the bandwidth available to TCP flows remained steady over time. However, we are aware that these assumptions are not generally valid and different situations may pose some limitations to the XFWA scheme. 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 the latter. 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 XFWA scheme.

One possible solution is to compute the returned feedback as follows:

$$\begin{aligned}&\mathrm{feedback}_i(0) = 1\\ &\mathrm{feedback}_i(n) = \mathrm{feedback}_i(n-1) + \frac{\mathrm{RTT}_i^{\alpha}}{\sum_{j=1}^{N}\mathrm{RTT}_j^{\alpha}}\\ &\qquad\cdot\left(Bw(n)\cdot \mathrm{RTT}_{\mathrm{avg}} + Q_{\mathrm{size}}(n)\right) \quad \forall n \geq 1\end{aligned} \tag{3}$$

where $\mathrm{feedback}_i(n)$ is the computed feedback of the ith flow at time $t = n\mathrm{RTT}_{\mathrm{avg}}$. $Q_{\mathrm{size}}(n)$ and $Bw(n)$ denote the free buffer size and the spare bandwidth both computed at time $t = n\mathrm{RTT}_{\mathrm{avg}}$. By so doing, when some connections are not making full use of their allocated bandwidths during the interval time $[(n-1)\mathrm{RTT}_{\mathrm{avg}}, n\mathrm{RTT}_{\mathrm{avg}}]$, the spare bandwidth will increase and the buffer occupancy will go down causing an increase in the computed feedbacks of the other connections at time $t = n\mathrm{RTT}_{\mathrm{avg}}$. This will help to fully utilize the link capacity while maintaining small buffer sizes. The authors are currently investigating the efficiency and fairness of XFWA when the feedback is computed according to the above equation.

VII. Conclusion

In this paper, we proposed an XFWA method to improve TCP performance over satellite networks. The method takes advantage of some specific attributes of multihops satellite constellations to make an approximate estimate of flows RTT and the bandwidth delay product of the network. To control network utilization, the scheme matches the sum of window sizes of all active TCP connections sharing a bottleneck link to the effective network bandwidth delay product. Min–max fairness, on the other hand, is achieved by assigning for each connection a weight proportional to its RTT. The computed feedbacks are signaled to TCP senders by modifying the RWND field carried by TCP ACKs. This operation can be accomplished without changing the protocol and, as a result, requires no modification to the TCP implementations in the end systems.

We demonstrated through extensive simulations that XFWA has the potential to substantially improve the system fairness, reduce the number of losses and makes better utilization of the link.

Experiments with dynamic changes in traffic demands showed that XFWA managed to control the buffer occupancy well and to achieve stability when a change in traffic load occurred. A large part of this success is due to the ability of XFWA to rapidly divide the available bandwidth among all active flows, while backing off the transmission rate of old flows upon arrival of new connections. Simulations with flows with different RTTs demonstrated the robustness of the XFWA to high variance in the flows RTT distribution and showed that XFWA is significantly fairer and has no bias against long RTT flows. Resiliency of the scheme to reasonable RTT estimation errors was demonstrated by a simple mathematical analysis and confirmed by simulation results. Abilities of the scheme to accommodate bursty traffic were verified also by considering a scenario where a mix of greedy and nonpersistent flows competes for the bandwidth of the bottleneck link.

From these experiment results, we believe that XFWA is a practical congestion control, specifically designed for multihops satellite constellations and represents a major contribution in TCP performance over satellite networks area.

Appendix I
Feedback Computation Load

This Appendix aims to demonstrate that XFWA is fairly simple to implement and does not require large computation work at the routers on board the satellites.

First, TCP flows are grouped according to the number of hops they traverse. Let κ denote the number of resulted groups. Let n_m and h_m denote the size of the mth group and the number of hops traversed by its flows, respectively, $(m \in [1, \kappa])$.

From (1), the estimated RTT value of flows of the mth group is

$$\mathrm{RTT}_m = 2\hbar_m \cdot \mathrm{ISL}_{\mathrm{delay}}$$

where $\hbar_m = h_m + 1$. From (2), the feedback of flows of the mth group can be expressed as[12]

$$\mathrm{Feedback}_m = \hbar_m^{\alpha} \cdot \Upsilon$$

where

$$\begin{aligned}\Upsilon &= \frac{\Re}{\Phi}\\ \Phi &= \sum_{j=1}^{\kappa} n_j \hbar_j^{\alpha}\\ \Re &= \mathrm{Bw}\cdot \mathrm{RTT}_{\mathrm{avg}} + Q_{\mathrm{size}}\\ \mathrm{RTT}_{\mathrm{avg}} &= 2\frac{\sum_{j=1}^{\kappa} n_j \cdot \hbar_j}{N} \cdot \mathrm{ISL}_{\mathrm{delay}}.\end{aligned}$$

[12]The feedback computation method is slightly modified. While in Section III-B, feedback computation is performed per flows for ease of explanation, the feedback is here computed per groups. Obviously, flows that belong to the same group will be provided with similar feedbacks.

N denotes the total number of TCP flows and is computed as

$$N = \sum_{j=1}^{\kappa} n_j.$$

Taking account of the $\mathrm{RTT}_{\mathrm{avg}}$, Φ, and $\Re$ computations and considering the case of $\alpha = 1$ (for the sake of simplicity), the computation of Υ gives rise to only two divisions, $2\kappa + 1$ multiplications and $2\kappa - 1$ additions. Since flows that belong to the same group have similar feedbacks, the periodic feedback computation of the whole system necessitates only κ multiplications in addition to the computation work of Υ.

On the other hand, the value of κ depends on the architecture of the satellite constellation and is always inferior than the maximum number of hops that can be traversed by any connection between any two terrestrial terminals. This threshold is denoted as χ. For instance, Wood [38] has showed through extensive simulation results that the longest one-way propagation delay experienced in the Teledesic constellation was less than 140 ms. Given that the ISL delay is 20 ms, the parameter χ can be assumed to be 6 in case of Teledesic constellation.

To conclude, the implementation of XFWA scheme is fairly simple and routers on-board XFWA satellites are not required to perform large work to compute feedbacks: maximum of two divisions, $3\chi + 1$ multiplications and $2\chi - 1$ additions. Furthermore, this computation work is not performed per packets or flows, but only once every $\mathrm{RTT}_{\mathrm{avg}}$ interval of time. In case of Teledesic, the maximum amount of computation load required at each XFWA router is merely 2 divisions, 19 multiplications, and 11 additions. Note that this amount of computation load is very practical even for high-speed routers.

APPENDIX II
RESILIENCY TO REASONABLE RTT ESTIMATION ERRORS

As described in Section III-B, the feedback computation is based on an approximate estimate of RTT. The following mathematical analysis shows that XFWA maintains good performance even for reasonable RTT estimation errors.

Let RTT_i and $\widehat{\mathrm{RTT}}_i$ denote the good estimate and erroneous estimate of RTT of the ith flow. e_i denotes the RTT estimation error ratio of the ith flow and is defined as follows:

$$e_i = \frac{\widehat{\mathrm{RTT}}_i - \mathrm{RTT}_i}{\mathrm{RTT}_i}.$$

Let $e_{\max}$ and $e_{\min}$ denote the maximum and minimum values of $(e_j, \forall j)$, respectively. From (2), the feedback value of the ith flow is

$$\mathrm{feedback}_i = \hat{\Psi}_i \cdot \hat{\Re}$$

where

$$\hat{\Psi}_i = \frac{\widehat{\mathrm{RTT}}_i^{\alpha}}{\sum_{j=1}^{N} \widehat{\mathrm{RTT}}_j^{\alpha}}$$
$$\hat{\Re} = \mathrm{Bw} \cdot \widehat{\mathrm{RTT}}_{\mathrm{avg}} + Q_{\mathrm{size}}$$
$$\widehat{\mathrm{RTT}}_{\mathrm{avg}} = \frac{\sum_{j=1}^{N} \widehat{\mathrm{RTT}}_j}{N}.$$

Assuming reasonable RTT estimation errors (i.e., $|e_i| \ll 1$) and using Taylor's approximation $((1+e_i)^{\alpha} \simeq 1 + \alpha e_i)$, $\hat{\Psi}$ can be approximated to

$$\hat{\Psi}_i = \frac{\Psi_i (1 + \alpha e_i)}{1 + \alpha \sum_{j=1}^{N} \Psi_j e_j}$$

where

$$\Psi_i = \frac{\mathrm{RTT}_i{}^{\alpha}}{\sum_{j=1}^{N} \mathrm{RTT}_j{}^{\alpha}}.$$

Using the following Taylor's inequality:

$$1 - x \preceq \frac{1}{1+x} \preceq 1 - x + x^2$$

and having $\sum_{j=1}^{N} \Psi_j = 1$, we obtain

$$1 - \alpha e_{\max} \preceq \frac{\hat{\Psi}_i}{\Psi_i (1 + \alpha e_i)} \preceq 1 - \alpha e_{\min} + (\alpha e_{\max})^2.$$

$(\hat{\Re} - \Re)$ can be also bound to

$$Bw \cdot \mathrm{RTT}_{\mathrm{avg}} e_{\min} \preceq (\hat{\Re} - \Re) \preceq Bw \cdot \mathrm{RTT}_{\mathrm{avg}} e_{\max}.$$

Assuming e_i to have a normal distribution with zero mean and variance $\sigma^2, (\hat{\Re} - \Re)$ also follows a normal distribution with zero mean and variance $\sigma_1^2 = (Bw^2/N^2) \cdot (\sum_{j=1}^{N} \mathrm{RTT}_j^2)\sigma^2$. In case of $\alpha = 1, e_{\max} = 0.15, e_{\min} = 0, \mathrm{RTT}_{\mathrm{avg}} = 160$ ms, $Bw = 1.544$ Mb, 1 pkt = 1 kB, we obtain

$$0.85 \preceq \frac{\hat{\Psi}_i}{\Psi_i} \preceq 1.175$$
$$0 \text{ pkts} \preceq (\hat{\Re} - \Re) \preceq 4.632 \text{ pkts}.$$

Note that the values of $\hat{\Psi}_i/\Psi_i$ and $(\hat{\Re} - \Re)$ should be in the vicinity of one and zero, respectively, so that the system's effectiveness would not be affected. This is possible in the case of reasonable RTT estimation errors; small values of $e_{\max}$ and $e_{\min}$. From the above mathematical analysis, we conclude that reasonable RTT estimation errors may have only marginal effects on the overall performance of the XFWA scheme.

APPENDIX III
SAME ROUTE FOR FORWARD AND BACKWARD TRAFFIC

Similarly to a number of schemes proposed in recent literature [23], [24], the XFWA scheme requires the forward traffic and the acknowledgments to traverse the same path. While implementation of routing approaches that help to solve this issue is outside the scope of this paper, this appendix purposes to clarify the possible cases in which this limitation might reduce the XFWAs effectiveness and suggests possible solutions.

In satellite networks, when the connection path is longer than one hop, more than one possible path between the end-systems can be simultaneously used. These multiple paths cause both data packets and acknowledgments to be received out of order

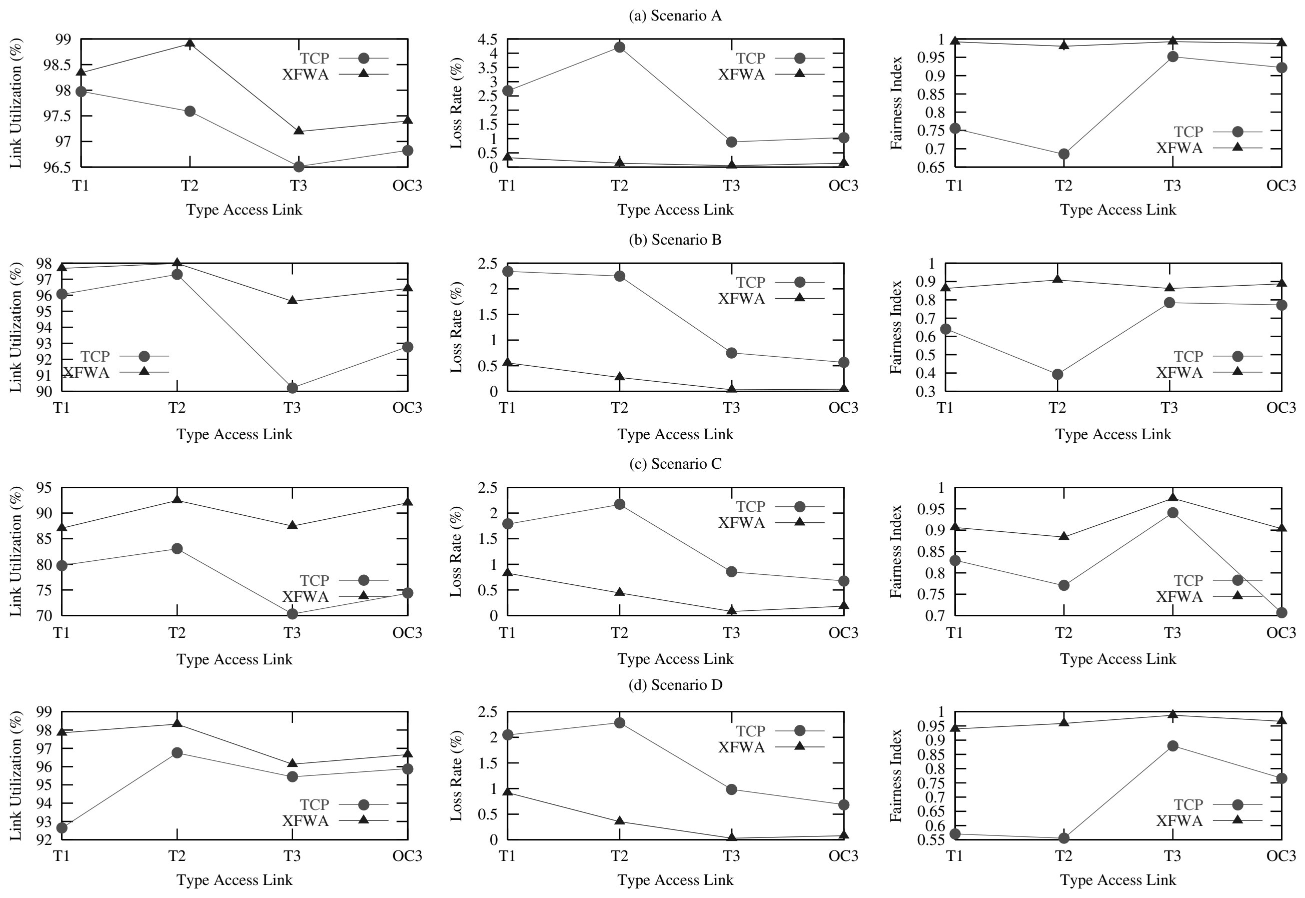

Fig. 17. Overall performance: Flows with different RTTs (skew factor $\alpha = 1$).

and eventually degrade the overall throughput of the connection. One suggested approach to tackling this problem of in-order delivery in multipath environments is the scheme proposed in [39]. The scheme measures the total propagation delays and queueing delays of these multiple paths, and uses this information to select an optimal and single path for each connection. Implementation of such a scheme can help XFWA satellites to compel forward traffic and the corresponding backward traffic to travel along the same route.

Another scenario where data packets and acknowledgments may travel along different paths and might eventually impose limitations on the performance of XFWA, occurs when either the destination or the source undergoes handover from a satellite to another. Upon handover occurrence, routing tables are updated, and packets that are still in transit to the satellite that was being used by the end-system (before handover occurrence), will be routed onward to the current satellite the end-system is now using. This will cause those "in-transit" packets to travel one extra hop and eventually results in an abrupt increase or decrease in the flow delay. As a consequence, erroneous estimates of flows RTT and the total number of flows may be obtained. Feedback values of all TCP flows might accordingly be affected. Nonetheless, since parameters are periodically updated, handover occurrence is most unlikely to coincide with the time of the update operation. Even if such a coincidence happens, feedbacks are periodically computed every $\mathrm{RTT}_{\mathrm{avg}}$ intervals of time and good estimates of parameters can be obtained in the next update interval of time. The effect of these "in-transit" packets will thus be minimal.

Appendix IV
Overall Network Performance in Case of Flows With Different RTTs

See Fig. 17 for the overall network performance in case of flows with different RTTs.

Acknowledgment

The authors would like to express their gratitude to the anonymous reviewers for their constructive comments and suggestions in improving this paper.

References

[1] D. E. Comer, *Internetworking with TCP/IP Principles, Protocols, and Architectures*, 4th ed. Englewood Cliffs, NJ: Prentice-Hall, 2000, vol. 1.

[2] I. F. Akyildiz, E. Ekici, and M. D. Bender, "MLSR: A novel routing algorithm for multilayered satellite IP networks," *IEEE/ACM Trans. Networking*, vol. 10, pp. 411–424, June 2002.

[3] S. Tekinay, *Next Generation Wireless Networks*. Norwell, MA: Kluwer, Nov. 2000.

[4] 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.

[5] N. Brownlee and K. C. Claffy, "Understanding Internet traffic streams: Dragonflies and tortoises," *IEEE Commun. Mag.*, vol. 40, pp. 110–117, Oct. 2002.

[6] C. Partridge and T. J. Shepard, "TCP/IP performance over satellite links," *IEEE Network*, vol. 11, pp. 44–49, Sept./Oct. 1997.
[7] M. Allman, S. Floyd, and C. Partridge, "Increasing TCP's initial window," Network Working Group, Internet RFC 2414, 1998.
[8] T. Henderson and R. Katz, "Transport protocols for Internet-compatible satellite networks," *IEEE J. Select. Areas Commun.*, vol. 17, pp. 326–343, Feb. 1999.
[9] 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.
[10] 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.
[11] 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.
[12] 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, pp. 307–321, June 2001.
[13] R. Morris, "TCP behavior with many flows," in *Proc. ICNP*, Atlanta, GA, Oct. 1997, pp. 205–211.
[14] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. Networking*, vol. 5, pp. 336–350, June 1997.
[15] G. Montenegro, S. Dawkins, M. Kojo, V. Magret, and N. Vaidya, "Long thin networks," Network Working Group, RFC 2757, Jan. 2000.
[16] L. Wood, G. Pavlou, and B. Evans, "Effects on TCP of routing strategies in satellite constellations," *IEEE Commun. Mag.*, vol. 39, pp. 172–181, Mar. 2001.
[17] 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.
[18] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, pp. 397–413, Aug. 1993.
[19] K. K. Ramakrishnan and S. Floyd, "Proposal to add explicit congestion notification (ECN) to IP," Network Working Group, RFC 2481, Jan. 1999.
[20] S. H. Low, F. Paganini, J. Wang, S. Adlakha, and J. C. Doyle, "Dynamics of TCP/AQM and a scalable control," presented at the INFOCOM, New York, June 2002.
[21] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End to end congestion avoidance on a global Internet," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1465–1480, Oct. 1995.
[22] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," presented at the SIGCOMM, Pittsburgh, PA, Aug. 2002.
[23] L. Kalampoukas, A. Varma, and K. K. Ramakrishnan, "Explicit window adaptation: A method to enhance TCP performance," *IEEE/ACM Trans. Networking*, vol. 10, pp. 338–350, June 2002.
[24] 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.
[25] A. K. Choudhury and E. L. Hahne, "Dynamic queue length thresholds in a shared memory ATM switch," in *Proc. INFOCOM*, Mar. 1996, pp. 679–687.
[26] L. Qiu, Y. Zhang, and S. Keshav, "On individual and aggregate TCP performance," in *Proc. ICNP*, Nov. 1999, pp. 203–212.
[27] D. M. Kohn, "Providing global broadband Internet access using low-earth-orbit satellites," *Comput. Networks ISDN Syst.*, vol. 29, no. 15, pp. 1763–1768, Nov. 1997.
[28] E. J. Fitzpatrick, "Spaceway system summary," *Space Commun.*, vol. 13, no. 1, pp. 7–23, 1995.
[29] 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. 241–251.
[30] R. Morris, "Scalable TCP congestion control," in *Proc. INFOCOM*, Mar. 2000, pp. 1176–1183.
[31] S. B. Fredj, S. O. Boulahia, and J. W. Roberts, "Measurement-based admission control for elastic traffic," presented at the ITC, Salvador, Brazil, Dec. 2001.
[32] Network Simulator-ns (Version 2) [Online]. Available: http://www.isi.edu/nsnam/ns/
[33] 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.
[34] S. Floyd and T. Henderson, "The NewReno modifications to TCP's fast recovery algorithm," Network Working Group, RFC 2582, Apr. 1999.
[35] D. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Compute Networks ISDN Syst.*, vol. 17, pp. 1–14, 1989.
[36] K. Park, G. Kim, and M. Crovella, "On the relationship between file sizes, transport protocols, and self-similar network traffic," in *Proc. Int. Conf. Network Protocols (ICNP)*, Columbus, OH, 1996, pp. 171–180.
[37] W. Willinger, M. Taqqu, R. Sherman, and D. Wilson, "Self-similarity through high variability: Statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Trans. Networking*, vol. 5, pp. 71–86, Feb. 1997.
[38] L. Wood, "Internetworking with Satellite Constellations," Ph.D. dissertation, Univ. Surrey, Surrey, U.K., 2001.
[39] M. L. Liron, "Traffic routing for satellite communication system," U.S. Patent 5 740 164, Apr. 1998.

Tarik Taleb received the B.E. and M.E. degrees in computer sciences from Tohoku University, Sendai, Japan, in 2001 and 2003, respectively. He is currently working toward the Ph.D. degree at the Graduate School of Information Sciences, Tohoku University.

His research interests lie in the field of wireless networking, especially in the performance of TCP over broadband satellite networks and satellite network security. His recent work has focused on on-demand media transmission in multicast environments.

Nei Kato (M'04) received the M.S. and Ph.D. degrees from the Graduate School of Information Sciences, Tohoku University, Sendai, Japan, in 1988 and 1991, respectively.

He joined the Computer Center, Tohoku University, in 1991 and is now a Professor with the Graduate School of Information Sciences. He has been engaged in research on computer networking, wireless mobile communications, image processing, and neural networks.

Dr. Kato is a member of the Institute of Electrical, Information, and Communications Engineers (IEICE).

Yoshiaki Nemoto (S'72–M'73) received the B.E., M.E., and Ph.D. degrees from Tohoku University, Sendai, Japan, in 1968, 1970, and 1973, respectively.

He is a Professor with the Graduate School of Information Sciences, and the Director of Information Synergy Center, Tohoku University. He has been engaged in research work on microwave networks, communication systems, computer network systems, image processing, and handwritten character recognition.

Dr. Nemoto was a corecipient of the 1982 Microwave Prize from the IEEE Microwave Theory and Techniques Society. He is a member of the Information Processing Society of Japan.