Understanding the Impacts of Short-Term Dynamics on Long-Term Fairness of Competing High Speed TCP Flows: A Root-Cause Analysis

Balázs Sonkoly, Tuan Anh Trinh and Sándor Molnár
High Speed Networks Lab, Department of Telecommunications and Media Informatics
Budapest University of Technology and Economics
H–1117, Magyar tudósok körútja 2, Budapest, Hungary
E-mail: {sonkoly, trinh, molnar}@tmit.bme.hu

***Abstract*—The short-term dynamics of competing high speed TCP flows can have strong impacts on their long-term fairness. This leads to severe problems for both the co-existence and the deployment feasibility of different proposals for the next generation networks. However, to our best knowledge, no root-cause analysis of the observation is available. In this paper, we try to fill this gap by providing an in-depth root-cause analysis of this phenomenon. We demonstrate that the widely used Jain's index as a fairness metric can not provide sufficient characterization of the phenomena. More precisely, Jain's index does not reflect the dynamic flow behaviors, e.g., starting time of the flows. We provide an analytical and simulation study to show the importance of the flow dynamics on fairness. We also propose a new metric called saturation time for fairness characterization. Both AIMD-based (HighSpeed TCP, BIC TCP) and MIMD-based (Scalable TCP) TCP versions are investigated in different topologies, namely dumb-bell and parking-lot topologies. In extreme cases, we also analyze and explain the "starving" effect of competing high speed TCP flows, when a flow forces other flows to deviate from their proper operation.**

I. Introduction

The advance in technologies, newer and newer forms of applications ensure improved and diversified services the end-users but they also bring new challenges for the network designers. In terms of bandwidth, with the emergence of fiber optical technologies, advance in multiplexing and modulation techniques such as OFDM and WDM, the networks can serve with the rate up to multiple gigabits/sec. In addition to bandwidth, network delay and delay variation are also important characteristics of a network. For example, transatlantic or satellite communication networks could have extremely high delay. The performance of these networks can be significantly different than network with average and low delay. A network that possesses both high bandwidth and large delay is usually called high bandwidth-delay product (BDP) network. Besides, bandwidth hungry and distributed applications such as peer-to-peer and multimedia applications often operate on these high BDP networks. As a result, there is a genuine need for next generation transport protocols that can efficiently utilize the resources and that can operate in these new and diverse network environments. This question has recently received considerable attention from the research community and a number of solutions have been proposed. Roughly, these protocols can be divided into two classes: loss-based and delay-based. Loss-based versions (e.g., HighSpeed TCP [1], Scalable TCP [2], BIC TCP [3], CUBIC [4], etc.) share similar features with traditional TCP (TCP Reno) whereas the delay-based ideas has resulted in FAST TCP [5] which is an extension of TCP Vegas. In most recent proposals such as Compound TCP [7], the combination of the delay-based and loss-based approaches has also been appeared. Other versions control the sending rate based on bandwidth estimation methods (e.g., TCP Westwood [8]). In this paper, we focus on loss-based versions of TCP that are designed as transport protocols for next generation networks characterized by high bandwidth-delay product. Both AIMD-based (HighSpeed TCP, BIC TCP) and MIMD-based (Scalable TCP) TCP versions are investigated in a wide range of network scenarios. One of the main reasons why we decide to revisit loss-based versions is because they share the same congestion control principles with the current TCP in use. Delay-based versions of TCP, such as TCP Vegas and FAST TCP, are promising when operating in homogeneous networks. However, there are serious concerns of their inter-operability with loss-based TCP proposals (such as TCP (New)Reno).

In order to understand the performance of the proposed protocols, benchmarking is needed. Benchmarking analysis of these new protocols is hard. Why? It is because there is no clear or general agreement on the set of the requirements for these protocols. It is however a status quo that one of the most important issues is operability and deployability. This directly leads to the question of fairness. In fact, this question is tackled by research community for quite a long time and a number of fairness metrics have been proposed, such as Jain's index, max-min fairness, proportional fairness, utility-based fairness, etc. These metrics are different, but they share a common aspect. They all concern with the long term average of the flows and their stable/equilibrium performance. The main weakness of these metrics is the lack of attention to the dynamic of the flows. It is argued and analyzed in this paper why and how starting time of the flows can have a great impact on the fairness of competing high speed TCP flows. We also analyze and explain the "starving" effect when a flow forces

other flows to deviate from their normal operation, in extreme cases, falling back to TCP Reno operation mode. This paper, as a part of our comprehensive fairness analysis [9], summarizes the main results on interaction of loss-based protocols.

The performance analysis of recently proposed mechanisms and TCP modifications is included in many papers. These works mainly deal with the performance of a new proposal or the interaction of standard TCP (Reno) and the new mechanism. In [10], [11], a simulation-based performance analysis of HighSpeed TCP is presented and the fairness to regular TCP is analyzed. In [2], the author deals with the performance of Scalable TCP and analyzes the aggregate throughput of the standard TCP and Scalable TCP based on an experimental comparison. In [5], the performance of different TCP versions, such as HighSpeed TCP, Scalable TCP, Linux TCP and FAST TCP, are compared in an experimental testbed environment. In all cases, the performance among connections of the same protocol sharing a bottleneck link is analyzed and different metrics are presented (throughput, fairness, responsiveness, stability). In [3], the authors compare the performance of BIC TCP using simulation with that of HighSpeed TCP, Scalable TCP and an AIMD mechanism. Bandwidth utilization, TCP friendliness, RTT unfairness, and convergence to fairness metrics are evaluated.

The analysis of competing flows using different TCP versions has received less attention. In [12], the fairness of MIMD algorithms is evaluated and the interaction of AIMD mechanisms with static parameters (e.g., TCP NewReno) are analyzed. In [13], an experimental evaluation of different high speed protocols, such as HighSpeed TCP, Scalable TCP, BIC TCP, FAST TCP and H-TCP, is presented. In a series of benchmark tests, the intra-protocol behavior of these TCP variants are analyzed considering the effect of starting time of the flows, as well. In [14], experimental evaluation of different high speed TCP proposals is carried out mainly focusing on the relevant impacts of background traffic. In [15], the intra-, and inter-protocol fairness of HighSpeed TCP, Scalable TCP, FAST TCP, H-TCP, BIC TCP and CUBIC is analyzed focusing on the impacts of starting time of the flows. The evaluation is based on simulations conducting in a simple dumb-bell topology with two competing flows. However, to our best knowledge, no root-cause analysis of the observation is available.

The need for creating a common performance evaluation framework for TCP versions has been identified and addressed by the IETF and IRTF working groups. In the internet draft [16] the metrics to be considered in an evaluation of new or modified congestion control mechanisms for the Internet has been collected. A benchmark tool has been presented in [17]. This benchmark consists of a set of network configurations (i.e., topologies, routing matrix, etc.), a set of workloads (i.e., traffic generation rules), and a set of metrics. The authors propose that the benchmark should be implemented in both Ns-2 simulation mode and hardware experiment mode, and they present some results from their on-going research. Another TCP evaluation suite has been suggested in the internet draft [18]. This consists an extendable tool that automates the Ns-2 TCP simulation process as much as possible. One can also define a set of commonly used network topologies, traffic models and performance evaluation metrics in the tool. A similar tool [19] has been developed based on an experiment scenario generator, consisting of a topology generator, a flows generator, and a workload generator, which are implemented in a set of tcl scripts for Ns-2 simulator.

The rest of the paper is organized as follows. In Section II, the motivation of our work is presented through simple examples. In Section III, the simulation environment and the important parameters are presented. In Section IV the transient and equilibrium behaviors of different TCP versions are analyzed and analytical results are derived. A novel metric (saturation time) is also introduced and derived for different TCP variants. Section V and VI present the main results of our comprehensive fairness analysis of competing high speed TCP flows according to our methodology. The impacts of starting delay are also examined and the explanations of the experienced phenomena are given, as well. Conclusions are drawn in Section VII.

II. Why is Jain's index insufficient?

One of the most popular and widely accepted fairness indices is Jain's index [20]. It is used widespread because of its main benefits [21]. Jain's index has a very important role in measuring fairness among large number of flows. It is a normalized metric being bounded between 0 and 1, and can be defined as follows: $JI = \left(\sum x_i\right)^2 / \left(n \sum x_i^2\right)$, where x_i is the normalized (e.g., average) throughput of the i-th flow and n is the number of flows. In contrast to other metrics, such as variance or standard deviation of throughput, it is independent of scale. Furthermore, it can be applied to any number of flows. Contrary to min-max ratio, it is continuous. And finally, this index has an intuitive relationship with user perception. Jain's index is particularly capable to describe the long term behavior of large number of flows. In other words, the static characteristics of the flow competition is captured especially.

Emerging networks bring new challenges. First, the new architectures, heterogeneous networks, mobile and wireless environments exhibit different network characteristics requiring more attention on the dynamical aspects of the operation. For example, in mobile environment, during an inter-system handover, the delay and the bandwidth can suddenly change. As for a TCP connection, these sudden changes in delay or the high value of jitter can cause multiple back-offs and, in extreme case, disconnection. Second, the network traffic is mainly determined by the popular applications. For example, web applications – generating a lot of short-time connections (dragonflies, mice traffic) – have a great importance today [22]. This type of traffic can not be treated considering only the long term properties.

As a consequence of the new network environments and properties, the dynamic behavior of the TCP flows must be taken into consideration. On the one hand, it is obvious that the dynamic effects have significant impact on the performance

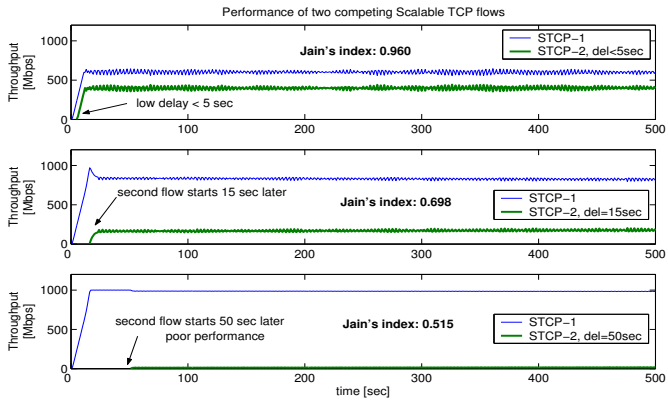


Figure 1. Performance of two competing Scalable TCP flows

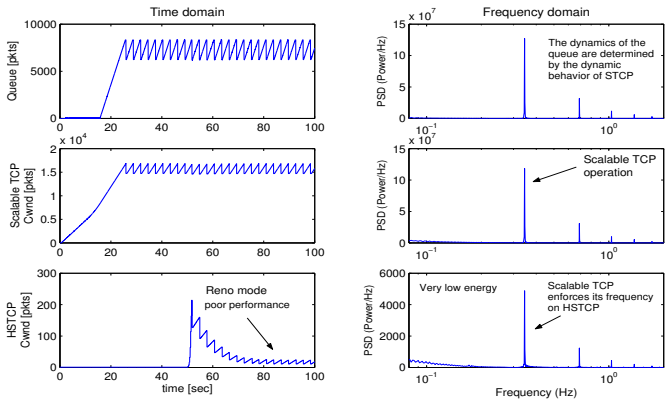


Figure 2. Competition of Scalable TCP and later entering HSTCP

and throughput of the TCP flows (see e.g., [23]). On the other hand, we argue that the fairness also needs to be reconsidered from the aspects of dynamic behavior. Jain's fairness metric is proposed assuming a simple control system model of n sources sharing the same bottleneck link and receiving the same feedback signal [20]. It can well describe the static properties of competing flows. However, the characteristics of the new network architectures and environments with new routing algorithms can not be well captured in all aspects by that model.

We show two simple examples – competition of two flows in the dumb-bell topology – to illustrate the deficiency of long term analysis and available fairness metrics. In case of two competing flows, we expect that the equilibrium properties and thus, the fairness in the sense of Jain's index, do not depend on the starting time of the flows. For example, a few seconds of starting difference between the flows can be omitted when the experiment lasts for a very long time (e.g., more than one hour). In Figure 1, the performance of two competing Scalable TCP flows is presented for three different starting delays. In the first case, the delay is less than 5 sec and the bandwidth shares are close to each other. This fact is also confirmed by Jain's index (0.960) approximating 1. The second scenario corresponds to a starting delay of 15 sec, and the bandwidth is shared less fairly which is reflected by a smaller Jain's index (0.698). The most surprising result can be observed in the last scenario. Here, the delay is increased to 50 sec and the second flow only achieves very low throughput. This unfairness is confirmed by Jain's index near to $1/2$. The importance of dynamic or transient analysis is illustrated by another example when the interaction of Scalable TCP and HSTCP can be observed. The long term average behavior of both flows can be approximated analytically and the average throughput can be expressed in terms of packet drop probability [6]:

$$x_{stcp} = \frac{1}{T}\frac{\alpha_{stcp}}{p} \quad \text{and} \quad x_{hstcp} = \frac{1}{T}\frac{\alpha_{hstcp}}{p^{0.84}},$$

where x_{stcp}, x_{hstcp} denotes the average throughput of a single Scalable TCP and HSTCP flow, respectively. T is an approximation of round-trip time at the equilibrium state and α_{stcp}, α_{hstcp} are constant parameters of the protocols. Thus, the equilibrium bandwidth share – assuming that the flows meet the same drop probability – can be expressed as follows:

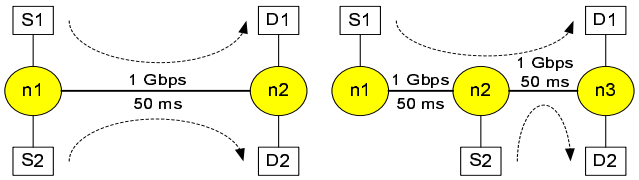
$$\frac{x_{stcp}}{x_{hstcp}} = \frac{\alpha_{stcp}}{\alpha_{stcp}}\frac{1}{p^{0.16}} \approx 0.625\,\frac{1}{p^{0.16}}. \qquad (1)$$

For small values of p, it can be expected that the Scalable TCP flow gets a slightly more bandwidth than HSTCP. However, the experiences show significantly different behavior and HSTCP is starved. *Dynamic analysis is needed in order to understand the interaction*. In Figure 2, the congestion window processes and the bottleneck queue are shown. (The simulation corresponds to the dumb-bell topology.) To our best knowledge, this is the first time to explain the poor performance of HSTCP when it co-exists with Scalable TCP. We have noticed that at the equilibrium, HSTCP source operates in Reno mode with very low values of congestion window and this is the root-cause of its poor performance. But why? We can answer this question by invoking one of the best tools for analysis in the frequency-domain: Fourier transform and FFT (Fast Fourier Transform). In Figure 2, beside the time-domain plots, the spectrum plots are depicted, as well. It can be observed that the dynamic properties of Scalable TCP is enforced on HSTCP – as the losses occur according to the main frequency spike of Scalable TCP. A HSTCP flow operating at this frequency can not leave Reno mode.

III. Simulation Environment

Our fairness analysis of competing high speed TCP protocols and the validation of the analytical results are carried out in the Ns-2 [24] simulation environment. Our simulation scripts regarding different network scenarios can be found in [9]. The different high speed transport protocols are integrated in the environment. Ns-2 version 2.27 includes the algorithm of HighSpeed TCP, while the Scalable TCP control mechanism can easily be implemented. The Ns-2 source code of BIC TCP is used from [25].

The examined dumb-bell topology containing one bottleneck link is shown in Figure 3a. The queueing mechanism corresponding to the bottleneck link is drop-tail. We do not consider the impacts of the buffer size in our analysis and the



(a) Dumb-bell (b) Simple parking-lot

Figure 3. Network topologies

Table I
PARAMETERS

Network parameters

capacity	1 Gbps
delay	50 ms
packet size	1,500 bytes

Buffer size

dumb-bell	8,333 pkts
parking-lot	25,000 pkts

HSTCP

Low_W	38
$High\_W$	83,000
$High\_P$	10^{-7}
$High\_Dec$	0.1

STCP

a	0.01
b	0.125

BIC TCP

β	0.8
S_{max}	32
S_{min}	0.01
B	4

buffers are set according to the bandwidth-delay product. We found that the quantitative properties of competing flows are affected by the size of the buffers in the network; however, the basic phenomena and the qualitative characteristics do not depend on this parameter. We also investigate a simple parking-lot topology (Figure 3b), where the impacts of different round-trip times (RTT) can be revealed. Here, only the second link is congested. In case of these scenarios, a simulation contains two competing flows starting at different time instances and performing an infinite FTP download. Investigating the impacts of the starting time, different values are chosen. More exactly, on the one hand, we analyze scenarios when the second flow enters later than the saturation time of the first flow (e.g., 50 sec delay), and on the other hand, scenarios with smaller delay (e.g., with 15 sec delay) are also examined. In the dumb-bell topology, the competition of a later entering flow against a traffic aggregate containing 10 flows using the same protocol is also analyzed.

During the evaluation, the default parameter set of the protocols is used (see [1] and [2]). HSTCP and Scalable TCP apply the Limited Slow-Start (LSS) mechanism [26], as well. The parameters of the simulations are summarized in Table I.

IV. TRANSIENT AND EQUILIBRIUM ANALYSIS OF TCP VERSIONS

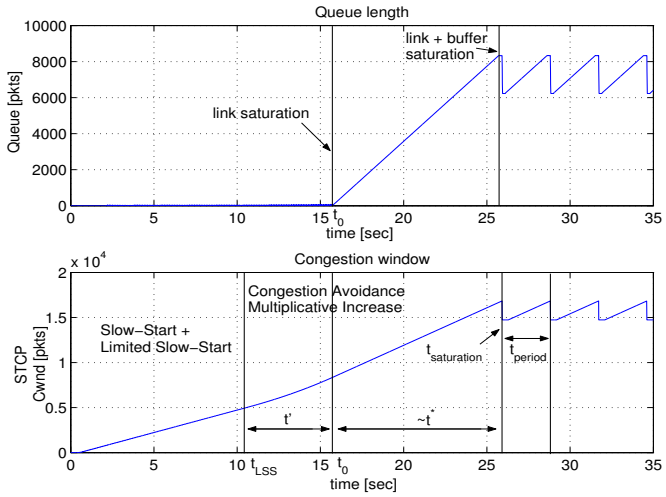
In this section, our recent analysis [9] on the behavior of individual flows is summarized in order to gain a basic knowledge of the behavior of different congestion control principles. Here, the investigation is carried out considering the simple dumb-bell topology. The performance of a single flow can be analyzed in two separate operating regimes. The first phase is a transient phase while the second one corresponds to an equilibrium behavior. We present our methodology through the example of Scalable TCP protocol. By this approach, other TCP variants can easily be treated and the important parameters can be derived analytically.

A. Initial dynamics – saturation time

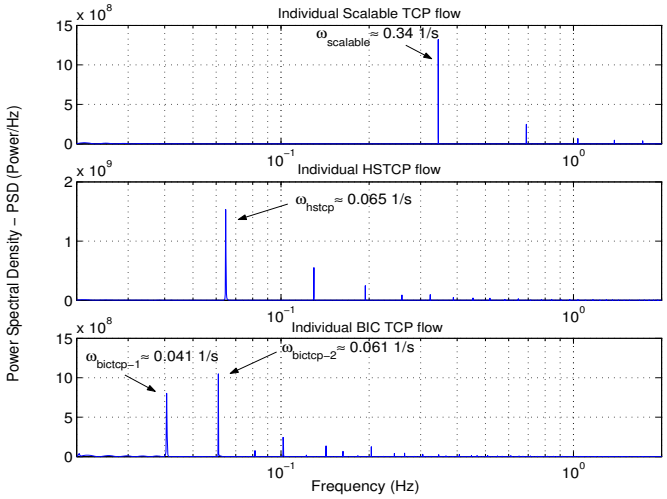
In this section, we focus on the initial phase which plays a significant role of the performance of an entering flow. We introduce a new performance metric, namely, the saturation time, as the length of this transient phase. This metric can be defined for a loss-based protocol as the time from the starting till the first packet drop. In Figure 4a, the saturation time and different phases of an individual Scalable TCP flow are presented as an illustration. Increasing the congestion window (and sending rate) of the source, the bottleneck link will be saturated after a while (link saturation). After this event, the buffer is filled by the new arriving packets. The time instance when the buffer is full at the first time is the saturation time.

Various TCP versions apply different mechanisms during the initial phase. A source generally starts sending according to a Slow-Start-like manner using a multiplicative increase algorithm with a protocol-dependent parameter. This means that the congestion window is increased by a constant value for each acknowledgement received. As an illustration, the saturation time of a Scalable TCP flow is derived. Our simulation results corresponding to this scenario are shown in Figure 4a with the main parameters. During the consecutive phases of initial operation, the Slow-Start, Limited Slow-Start (LSS) and the multiplicative increase mechanism of Scalable TCP are applied. In [9], we summarize the main characteristics of these analytically tractable control algorithms and we derive the relevant parameters, as well. In our scenarios, the Slow-Start phase is left when the initial threshold ($\texttt{ssthresh} = 100$ pkts) is exceeded. This time instance can easily be expressed as $t_{SS} = R_0 \log_2 \texttt{ssthresh} \approx 0.664$ sec, where R_0 is the round-trip propagation delay. After t_{SS}, the source operates according to the LSS mechanism using the default parameter ($\texttt{max\_ssth} = 100$ pkts). LSS operates in congestion avoidance mode in the Ns-2 implementation till the first packet drop. It affects the increase mechanism of `cwnd` comparing the increment of the congestion control mechanism (e.g., Scalable TCP, HSTCP) with its own increment and the maximum of these values are used. With this algorithm, a faster convergence can be achieved when the source sending rate is far from the equilibrium value. In Limited Slow-Start phase, `cwnd` is increased by at most $\texttt{max\_ssth}/2$ per round-trip time. The end of the LSS phase, actually, can be caused by a packet drop or the fact that the protocol's increase mechanism "suggests" more aggressive increment than the LSS algorithm. In our simulations, the end of this phase depends on the protocols and other network parameters, as well. In case of Scalable TCP, the end of LSS phase can be expressed as follows (see [9] for details):

$$\begin{aligned} t_{LSS} &= R_0 \frac{\lg \texttt{max\_ssth}}{\lg 2} + R_0 \frac{W_{LSS} - \texttt{max\_ssth}}{\texttt{max\_ssth}/2} \\ &\approx 10.46 \,\text{sec}, \end{aligned} \tag{2}$$



(a) Saturation time of Scalable TCP



(b) Spectrum of `cwnd` process of individual flows

Figure 4. Transient and equilibrium behavior

where W_{LSS} is the value of congestion window triggering the end of LSS. (For Scalable TCP, $W_{LSS} = 5,000$ pkts.) After Limited Slow-Start, the multiplicative increase mechanism of the protocol operates. During this period, the congestion window is increased from W_{LSS} to the BDP (R_0C). Thus, the link saturation time can easily be determined:

$$t_0 = t_{LSS} + t' = t_{LSS} + R_0 \frac{\lg \frac{R_0C}{W_{LSS}}}{\lg(1+a)} \approx 15.6\,\text{sec}. \quad (3)$$

The time till the first packet drop can also be determined by solving differential equations describing the dynamics of congestion window and the behavior of the queue. Instead of solving complicated differential equations (with varying delays and recursive arguments), a simple approximation can be applied. In this phase, the congestion window is increased from $W_0 = R_0C$ to $R_0C + B$ according to the multiplicative increase mechanism. Approximating the increase of the queueing delay by a linear function, the round-trip time can be treated as a constant with a mean value: $\tilde{R} = R_0 + B/2C$. Thus, the saturation time can be expressed as follows:

$$\hat{t}_{saturation} = t_0 + t^* = t_0 + \tilde{R} \frac{\lg \frac{R_0C+B}{R_0C}}{\lg(1+a)} \approx 26.05\,\text{sec}. \quad (4)$$

The analytically derived parameters and the approximation of saturation time meet well the simulation results presented in Figure 4a. In Table II, we summarize our results on the transient behavior of different protocols (for details, see [9]).

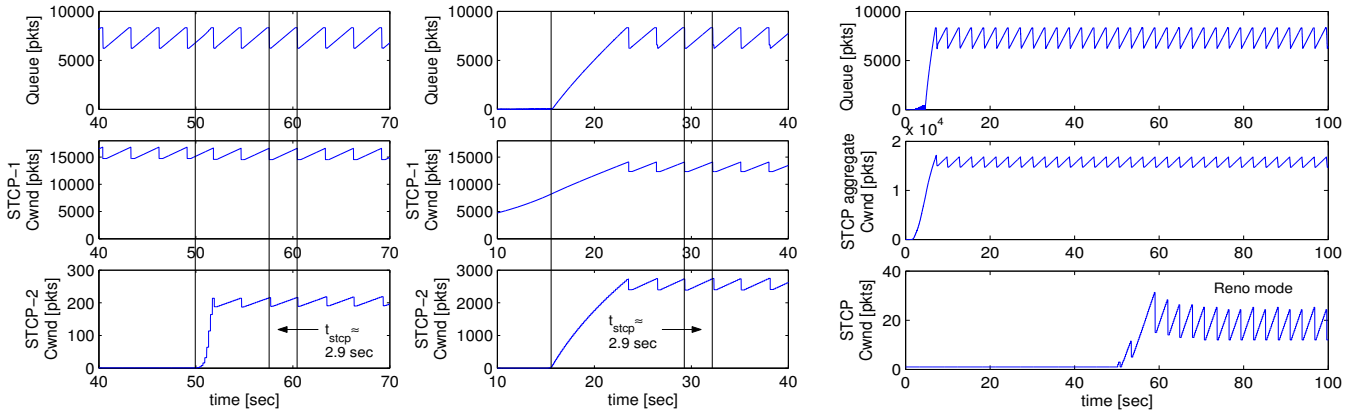
B. Equilibrium behavior

After the saturation time, an individual flow using a loss-based protocol shows periodic equilibrium behavior with periodic packet losses. The relevant parameters characterizing this state can also be derived analytically for the protocols, respectively. To understand the long term behavior of individual flows is crucial in order to understand the interaction of different flows later. In this section, we summarize the long term characteristics of the examined protocols. Further details and the analytical derivations can be found in [9].

Table II
APPROXIMATION OF SATURATION TIME OF DIFFERENT PROTOCOLS

STCP:	$t_{LSS} + R_0 \frac{\lg \frac{R_0C}{W_{LSS}}}{\lg(1+a)} + \tilde{R} \frac{\lg \frac{R_0C+B}{R_0C}}{\lg(1+a)}$	≈ 26 sec
HSTCP:	$t_{SS} + R_0 \frac{R_0C - \texttt{max\_ssth}}{\texttt{max\_ssth}/2} + \tilde{R} \frac{B}{\texttt{max\_ssth}/2}$	≈ 42 sec
BIC TCP:	$R_0 \frac{\lg \texttt{max\_ssth}}{\lg 2} + R_0 \frac{\lg \frac{R_0C}{\texttt{max\_ssth}}}{\lg(1+a)} + \tilde{R} \frac{\lg \frac{R_0C+B}{R_0C}}{\lg 1+a}$	≈ 12 sec

As a simple illustration, we present the long term behavior of an individual Scalable TCP flow. The MIMD mechanism of the protocol – operating during the equilibrium state – yields a time period $k = -\log(1-b)/\log(1+a)$ (expressed in RTT). After the first packet drop, the Scalable TCP source operates around an operating point when the bottleneck queue is approximately full. Thus, the round-trip time ($R(t)$) can be approximated at that operating point by $R(t) \approx \tilde{R} = R_0 + B/C$, where R_0 is the round-trip propagation delay, C is the bottleneck capacity, and B is the buffer length. In our example $\tilde{R} \approx 0.2$ sec. According to these results, the time of a period (t_{period}) can be approximated by $k\tilde{R} \approx 2.6 \ldots 2.8$ sec. The analytical result well captures the periodic behavior experienced in the simulations. In Figure 4a, the length of a period $t_{period} \approx 2.9$ sec. This period also contains the time which is needed for retransmit and recovery. The spectrum of the congestion window process can also de derived. The relevant part of the spectrum is shown in Figure 4b (top). The main spike corresponding to the dominant frequency can be seen at approximately $\omega = 0.341/s$ which meets well the



(a) 2 flows, delay: 50s (b) 2 flows, delay: 15s (c) individual flow vs. traffic aggregate

Figure 5. Intra-protocol behavior: Scalable TCP

equilibrium time period derived analytically.

For the other loss-based protocols, the periodic characteristics can be determined in a similar way. The details of analytical derivations can be found in [9]. In Figure 4b, we show the spectrum plots of the congestion window processes of individual flows. The main frequency spike of HSTCP can be seen at approximately $\omega = 0.065\,1/s$ which is significantly smaller than the frequency of Scalable TCP. The control mechanism of BIC TCP yields two main spikes at $0.04\,1/s$ and $0.06\,1/s$, respectively.

V. Intra-protocol behavior

As an essential requirement, a TCP protocol should guarantee fair behavior among flows using that protocol. The next step of our investigation involves the analysis of the interaction of the same TCP flows in order to get a basic knowledge of the network behavior when all the sources use the same transport protocol. This topic has received more attention recently, and a lot of results can be found in the literature (see e.g., [5], [13]). Therefore, the aim of this section is rather to get a basic knowledge and to confirm and explain certain results than to provide a comprehensive study. The details of our analysis can be found in [9]. Here, we summarize only our main findings and the main properties of the intra-protocol behavior for different types of loss-based congestion control principles.

A. MIMD mechanism

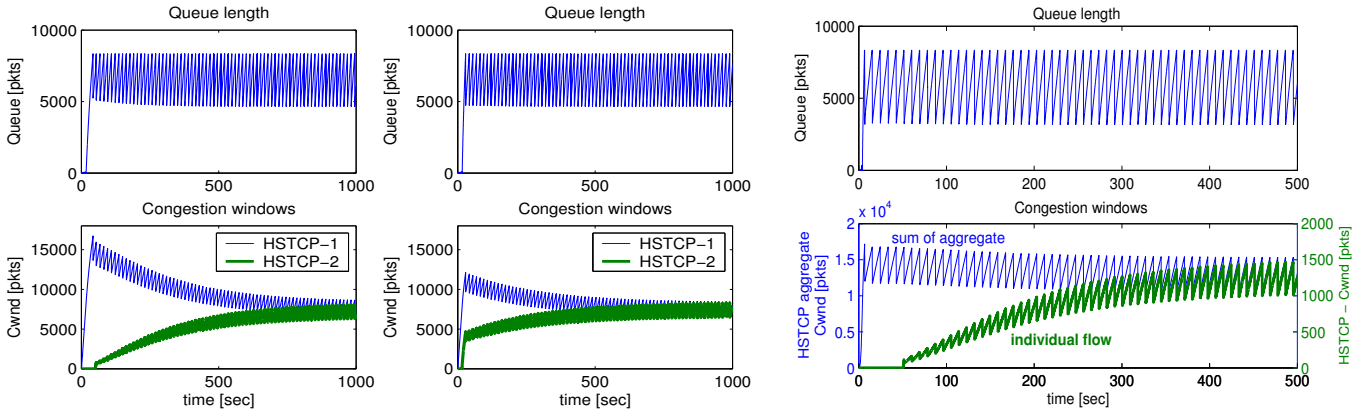
Our first statement is the following: *the MIMD mechanism can not guarantee the fair behavior among flows using the same MIMD algorithm assuming synchronized losses even in very simple network environment.* More exactly, the performance of the competing Scalable TCP flows are mainly affected by the starting time of the flows.

As an illustration, the competition of two Scalable TCP flows in a very simple dumb-bell topology is presented. The congestion window processes and the dynamics of the bottleneck queue are shown in Figure 5a and Figure 5b for two different starting delays. In the first scenario, the second flow enters the network after the saturation time of the first one (50 sec), while the second scenario corresponds to a delay (15 sec) smaller than the saturation time. As a consequence of the properties of the MIMD algorithm, during the equilibrium phase, the two sources operate at the same frequency. The performance of the second flow and the fairness are mainly determined by the state of the first flow at the time instance of entering. Thus, the synchronized losses and synchronized periods of the two Scalable TCP flows can cause an unfair equilibrium state and unfair bandwidth share when the second flow is starved. Moreover, a later entering Scalable TCP flow shows very poor performance competing with a traffic aggregate of Scalable TCP flows. The queue process and the congestion window process of the traffic aggregate (as the sum of the individual processes) and the late entering flow are shown in Figure 5c. Here, the competition of a single flow against a traffic aggregate of 10 flows are presented. Starting times of the traffic aggregate flows are uniformly distributed within the first 5 sec while the last flow enters 50 sec later. Because of the basic properties of the MIMD algorithm, the dominant frequency of the traffic aggregate equals the one derived for a single Scalable TCP flow (see Figure 4b). The main frequency spike can be observed at approximately $0.34\,1/s$ in both cases. The later entering flow shows very poor performance operating in Reno mode which is a *serious disadvantage* of the protocol from practical aspects.

B. AIMD-like mechanisms

The less aggressive congestion control schemes using additive or slower (logarithmic) increase mechanisms are able to realize fair equilibrium states for similar flows; however, the transient phases can last unacceptable long time. In case of HSTCP and BIC TCP, the starting time has an impact on this transient phase but the long term behavior is not affected.

As an illustration, we present some results on the intra-protocol behavior of HSTCP. The adaptive nature of the protocol originates from considering the current value of the



(a) 2 flows, delay: 50s (b) 2 flows, delay: 15s (c) individual flow vs. traffic aggregate

Figure 6. Intra-protocol behavior: HSTCP

congestion window during the "rate" adjustment. In Figure 6, the dynamics of congestion windows and the bottleneck queue are shown for three different scenarios using the dumb-bell topology. These scenarios are similar to the previously presented ones for Scalable TCP (two flows – 50 sec delay, two flows – 15 sec delay, traffic aggregate – individual flow). Our results show that *in case of HSTCP flows operating in simple dumb-bell network environment, the starting delay has an impact on the convergence time; however, the long term fairness is not affected and a fair equilibrium state is realized*. In Figure 6c, the vertical axes of the congestion window plot corresponding to the traffic aggregate (left axis) and the individual flow (right axis) are scaled differently. A fair equilibrium state is realized after a quite long transient period. (The ratio of the congestion window of the individual flow and the sum of the congestion windows of the aggregate is approximately $1:10$.)

VI. Interaction of TCP versions

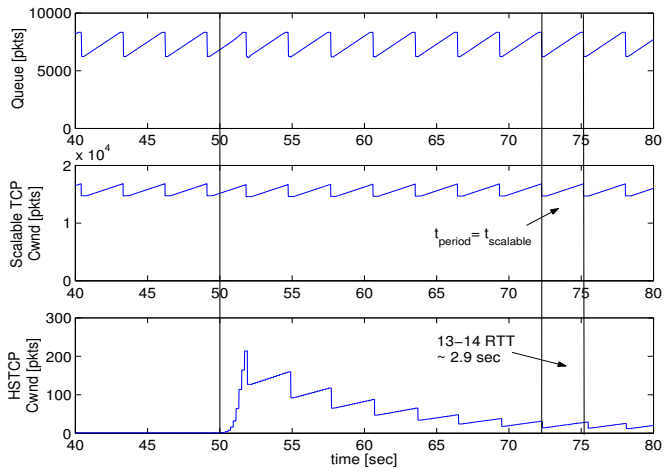
Today, it is an important question that how the recently proposed transport protocols can *live* beside each other in a shared network environment. An essential part of a loss-based high speed transport protocol is the adequate increase mechanism. On the one hand, the multiplicative increase algorithm increases the congestion window by a constant value for each acknowledgement independently the current value of the window. On the other hand, adaptive additive increase mechanisms can change the increment according to the current value of the congestion window. (The exact parameters and the dynamic nature are different for these AIMD-like protocols.) In this section, the interaction of the two different congestion control principles (AIMD-like and MIMD mechanisms) are investigated based on the analysis of a diverse set of scenarios.

1) Dumb-bell topology: single flows: To reveal the basic properties of the interaction, first we investigate the competition of two single flows in a simple dumb-bell topology.

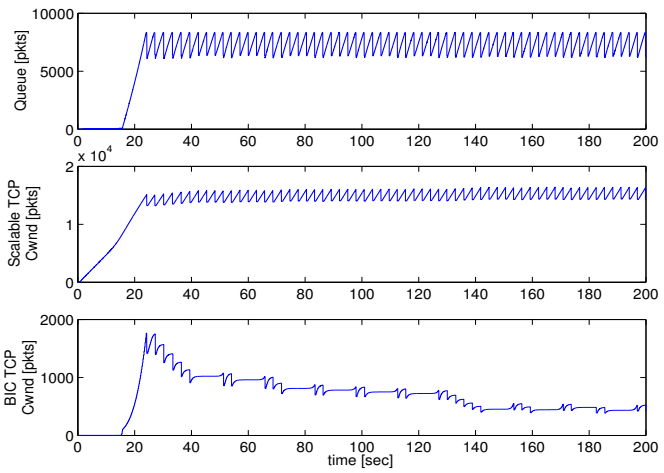
A surprising phenomenon can be observed when a HSTCP source enters the network after a Scalable TCP flow has achieved its maximal sending rate. Our first example is shown in Figure 7a. Here, the congestion window processes and the dynamics of the bottleneck queue are presented. In this scenario, the HSTCP source starts sending after the saturation time of the Scalable TCP flow (the delay is 50 sec exactly). HSTCP starts sending according to the Slow-Start algorithm and the first packet drop occurs synchronized with the other flow and triggers the congestion avoidance phase. Losses (caused by buffer saturations) occur synchronized between the two flows during the connection. The periodic behavior of HSTCP is exactly determined by Scalable TCP and a common time period is exhibited. (The spectrum plots confirming that are shown in Figure 2.) In our simulation example, the time period of Scalable TCP is approximately $13-14$ RTT. During this time interval, HSTCP can achieve a maximum increment of $13\,a(W)$ of `cwnd`, while the decrement ($b(W)$) is greater yielding a decreasing trend. In the equilibrium state, `cwnd` will be smaller than Low_Window parameter of HSTCP which results in *TCP Reno operating mode* ($a(W)=1$, $b(W)=0.5$) and poor performance. During the equilibrium state, Scalable TCP forces the HSTCP flow to deviate from its normal operation. The starting time of HSTCP affects only the length of the transient phase. Moreover, when HSTCP starts before the saturation time of Scalable TCP or the sources start at the same time, the equilibrium states are the same, only the length of the transient time differs.

The interaction between Scalable TCP and BIC TCP shows similar characteristics when the Scalable TCP flow starts first. The later entering BIC TCP flow can not achieve significant rate; however, the performance is not as poor as in the previous case ("non-Reno mode"). As an illustration, the `cwnd` processes and the queue dynamics are shown for 15 sec starting delay in Figure 7b. The equilibrium state does not depend on the starting time of the BIC TCP flow.

A better performance is expected when the Scalable TCP enters later the network. Surprisingly, the results do not meet the expectations and the starvation of AIMD-like flows can be observed in these cases, too. As an example, we examine the interaction of a HSTCP flow and a 50 sec later entering



(a) Scalable TCP – HSTCP, delay: 50 sec



(b) Scalable TCP – BIC TCP, delay: 15 sec

Figure 7. Inter-protocol behavior: Scalable TCP – other loss-based protocol using AI mechanism

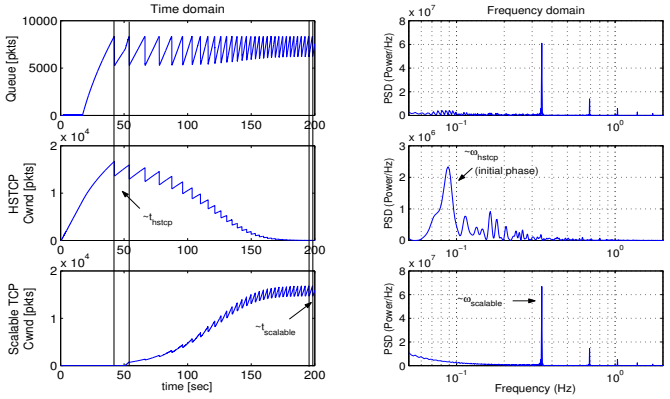


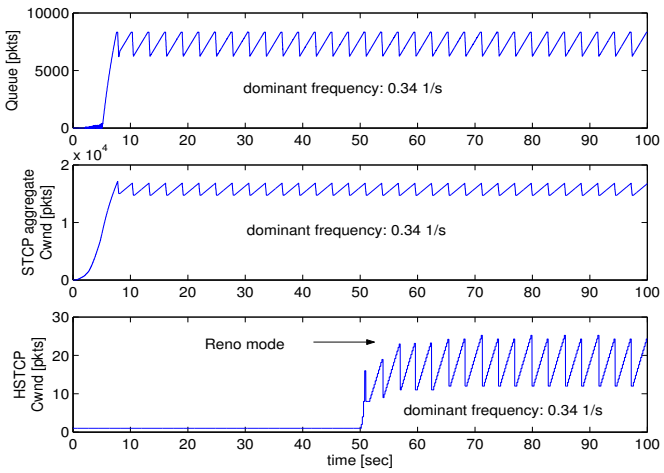
Figure 8. HSTCP – Scalable TCP, delay: 50s

Scalable TCP flow. The simulation results are shown in Figure 8. At the time of starting Scalable TCP, HSTCP has achieved its equilibrium state with a time period of t_{hstcp}. Scalable TCP starts with Slow-Start/Limited Slow-Start and the bottleneck queue is fed by the traffic aggregate of the two flows. The extra traffic of Scalable TCP in the queue results in a decreasing time period, i.e., the synchronized losses occur more frequently. During shorter time periods, `cwnd` of HSTCP can not achieve the value that was just before the reduction. In contrast with HSTCP, `cwnd` of Scalable TCP – adjusted according to the multiplicative increase algorithm – exceeds the value that has been reached before the reduction at the end of a period. Thus, the length of a period converges to the time period of Scalable TCP ($t_{scalable}$) and `cwnd` of HSTCP shows a decreasing trend while `cwnd` of Scalable TCP is increasing. The equilibrium state is the same as it was experienced previously. The same phenomena can also be observed in the frequency-domain. In Figure 8, the spectrum plots are shown beside the time-domain plots. The diagrams confirm that the long term network behavior is mainly determined by the Scalable TCP flow, since the bottleneck queue shows the same dominant frequency as the MIMD mechanism.

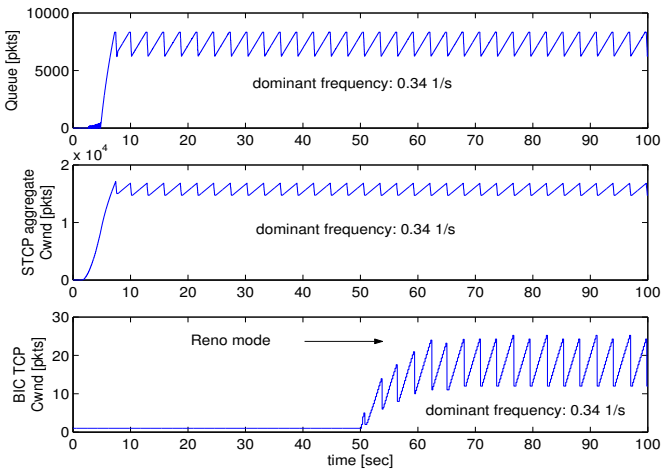
In case of BIC TCP, the phenomena are similar. The long term performance and the equilibrium behavior are the same when the BIC TCP flow starts first or later. It is worth noting, that BIC TCP achieves better utilization than HSTCP because it can leave Reno mode.

2) Dumb-bell topology: traffic aggregate – single flow: After the investigation of two interacting flows, this section deals with more realistic scenarios when one single flow competes with a traffic aggregate using the other type of congestion control mechanism. The simulation results are presented when the late entering flow competes with 10 other flows operating in their equilibrium state. (The starting times of the traffic aggregate flows are uniformly distributed within the first 5 sec.)

Our first observation is that the later entering AIMD-like flow can not achieve significant bandwidth share against the Scalable TCP aggregate and HSTCP and BIC TCP operate as TCP Reno (`cwnd` $<$ 30). The bottleneck queue and the `cwnd` processes corresponding to the two scenarios are shown in Figure 9. The upper parts of the figures relate to the queuing processes while the lower parts correspond to the traffic aggregates and the individual flows. In case of the aggregate, the data is the sum of the data series of the individual congestion windows. The periodic behavior of the individual AIMD-like flow is exactly determined by the MIMD mechanism of Scalable TCP, since the spectrum of HSTCP and BIC TCP shows the same dominant frequency spikes. (Here, we only present the dominant frequencies and the spectrum plots are omitted. Further analysis can be found in [9].) As a consequence of the MIMD algorithm, the spectral behavior of the Scalable TCP aggregate is similar to the behavior of an individual flow.

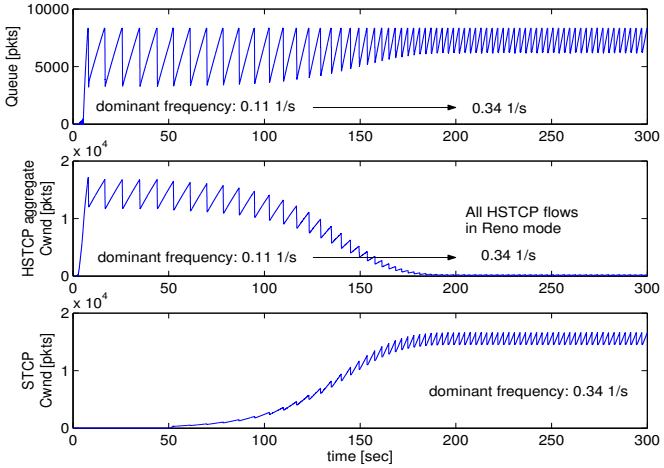


(a) STCP aggregate – HSTCP

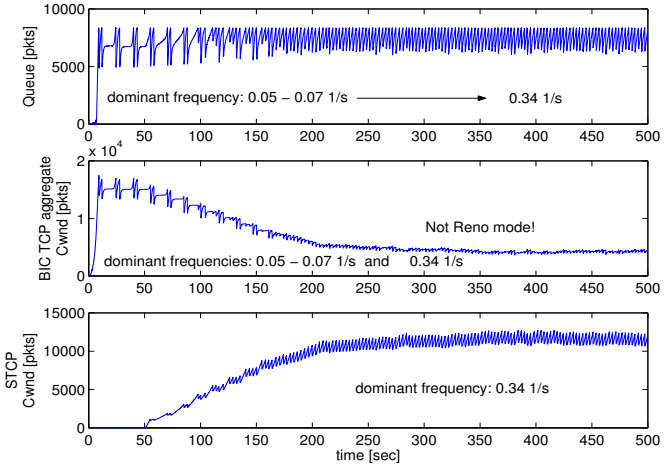


(b) STCP aggregate – BIC TCP

Figure 9. Performance of individual flow vs. STCP traffic aggregate



(a) HSTCP aggregate – STCP



(b) BIC TCP aggregate – STCP

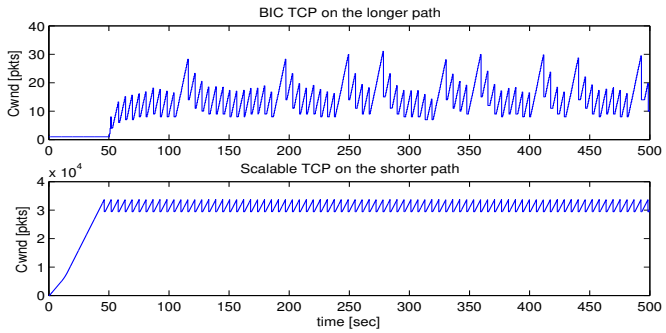
Figure 10. Performance of individual STCP flow vs. traffic aggregate

The next disadvantage of Scalable TCP is shown in Figure 10. Surprisingly, the individual Scalable TCP flow can starve the HSTCP and BIC TCP aggregates, as well. On the one hand, HSTCP flows operate in Reno mode achieving very low utilization at the equilibrium state and their periodic behavior is determined by the single Scalable TCP flow. On the other hand, the performance of the BIC TCP aggregate is better ("non-Reno mode"); however, the dominance of the Scalable TCP flow is significant.

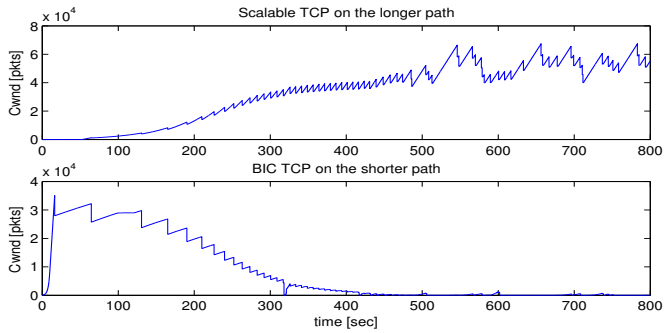
3) Simple parking-lot topology: single flows: Finally, the interaction of the MIMD and AIMD-like protocols is analyzed in a simple parking-lot topology (with one congested link). In these scenarios, the impacts of the different round-trip times can be revealed. When the Scalable TCP flow possesses the shorter RTT, the starving of the AIMD-like flows is expected. As an illustration, the interaction of a single Scalable TCP flow and a later entering BIC TCP flow with longer RTT is shown in Figure 11a. The BIC TCP flow operates in Reno mode achieving very low sending rate. The long term behavior is similar for other scenarios, too, and the AIMD-like flows are starved operating in Reno mode. The starting time has an impact only on the transient characteristics.

An important property of Scalable TCP arises when it traverses the longer path. An illustrative result is shown in Figure 11b. Here, the later entering Scalable TCP flow forces down the BIC TCP flow. As a result of the aggressive nature of the MIMD algorithm, the AIMD-like flows are always starved on the shorter path, too. However, the performance is slightly better than in Reno mode.

4) Results: Our main findings are the following. The MIMD mechanism can not guarantee the fair behavior with AIMD-like protocols using adaptive additive increase algo-



(a) BIC TCP on the longer path starts later



(b) BIC TCP on the shorter path starts first

Figure 11. Simple parking-lot topology: interaction of Scalable TCP and BIC TCP

rithm. Because of the aggressive and static characteristics of the multiplicative increase control of Scalable TCP, it starves other flows with AIMD protocols in a wide range of network environments. The aggressive nature of the MIMD mechanism is exhibited in the parking-lot topology and in the competition against traffic aggregates, as well. We found that the long term interaction of these protocols is not affected by the starting delay. Starting delay has an impact on the transient phase and the convergence time.

VII. Conclusion

In this paper we have presented a root-cause analysis of the fairness behavior of both AIMD-based (HighSpeed TCP, BIC TCP) and MIMD-based (Scalable TCP) TCP versions in both dumb-bell and parking-lot topologies. We have shown that the fairness metrics proposed and used so far (e.g., Jain's index) cannot capture the dynamic characteristics of the TCP flows. On the other hand, we have clearly pointed out that TCP flow dynamics do have significant impact on the long-term fairness performance.

We have proposed the starting time as a complementary dynamic-sensitive metric to the list of the long-term equilibrium sensitive metrics to get a full fairness characterization.

We have derived both analytical and simulation results in different scenarios including both inter-protocol and intra-protocol settings. In addition we have used spectral analysis to explain some unexpected co-working behavior of different TCP versions and show how a TCP flow can force other flows to deviate from their normal operation.

In the future we plan to continue the analysis and our aim is to contribute with new metric-related results to the congestion control performance evaluation framework which is presently being developed by the IETF.

References

[1] S. Floyd, HighSpeed TCP for Large Congestion Window, *RFC 3649*, December 2003.

[2] T. Kelly, Scalable TCP: Improving Performance in Highspeed Wide Area Networks, *ACM SIGCOMM Computer Communication Review*, 33(2), April 2003.

[3] L. Xu, K. Harfoush, I. Rhee, Binary Increase Congestion Control for Fast Long-Distance Networks, in *Proc. of IEEE INFOCOM*, 2004.

[4] I. Rhee, L. Xu, CUBIC: A New TCP-Friendly High-Speed TCP Variant, *PFLDnet*, 2005.

[5] C. Jin, D. X. Wei, S. H. Low, FAST TCP: motivation, architecture, algorithms, performance, in *Proc. of IEEE INFOCOM*, March 2004.

[6] D. X. Wei, C. Jin, S. H. Low, S. Hegde, FAST TCP: motivation, architecture, algorithms, performance, *IEEE/ACM Transactions on Networking (TON)*, v.14 n.6, p.1246-1259, December 2006.

[7] K. Tan, J. Song, Q. Zhang, M. Sridharan, A Compound TCP Approach for High-speed and Long Distance Networks, in *Proc. of IEEE INFOCOM*, April 2006.

[8] R. Wang, K. Yamada, M. Y. Sanadidi, M. Gerla, Tcp with sender-side intelligence to handle dynamic, large, leaky pipes, in *IEEE Journal on Selected Areas in Communications*, 23(2):235- 248, February 2005.

[9] B. Sonkoly, T. A. Trinh, S. Molnár, Benchmarking High Speed TCP Fairness, *Technical Report BME*, Oct. 2007. http://hsnlab.tmit.bme.hu/projects/tcp/

[10] E. Souza, D. Agarwal A HighSpeed TCP Study: Characteristics and Deployment Issues, *Lawrence Berkeley National Lab's Technical Report*, Berkeley, USA, 2003.

[11] T. A. Trinh, B. Sonkoly, S. Molnár, A HighSpeed TCP Study: Observations and Re-evaluation, *IFIP EUNICE*, 2004.

[12] E. Altman, K.E. Avrachenkov, B.J. Prabhu, Fairness in MIMD Congestion Control Algorithms, in *Proc. of IEEE INFOCOM*, March 2005.

[13] Yee-Ting Li, Douglas Leith, Robert N. Shorten, Experimental evaluation of TCP protocols for high-speed networks, *IEEE/ACM Transactions on Networking*, v.15 n.5, p.1109-1122, October 2007.

[14] S. Ha, Y. Kim, L. Le, I. Rhee, L. Xu, A Step toward Realistic Performance Evaluation of High-Speed TCP Variants, *PFLDnet*, 2006.

[15] M.C. Weigle, P. Sharma, and J. Freeman, Performance of Competing High-Speed TCP Flows, in *Proc. of IFIP Networking*, May 2006.

[16] S. Floyd, Metrics for the Evaluation of Congestion Control Mechanisms, draft-irtf-tmrg-metrics-09.txt, March 2007.

[17] D. X. Wei, P. Cao, S. H. Low, Time for a TCP benchmark suite? Available from: http://www.cs.caltech.edu/ weixl/research/technicals/ benchmark/summary.ps

[18] G. Wang, Y. Xia, D. Harrison, An NS2 TCP Evaluation Tool Suite, draft-irtf-tmrg-ns2-tcp-tool-00, April 2007.

[19] TCP Evaluation Suite Available from: http://netlab.cs.ucla.edu/tcpsuite/

[20] D. Chiu, R. Jain, Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks, *Computer Networks and ISDN Systems*, pages 1-14, June 1989.

[21] R. Jain, A. Durresi, G. Babic, Throughput Fairness Index: An Explanation, ATM_Forum/99-0045, February 1999, http://www.cse.wustl.edu/ jain/atmf/a99-0045.htm

[22] N. Brownlee, K. Claffy, Understanding Internet Traffic Streams: Dragonflies and Tortoises, *IEEE Communications Magazine*, 40(10):110–117, October 2002.

[23] A. Gurtov, Effect of Delays on TCP Performance, in *Proc. of IFIP Personal Wireless Communications 2001*, Lappeenranta, Finland, 2001.

[24] Ns-2 Network Simulator, Obtain via http://www.isi.edu./nsnam/ns.

[25] BI-TCP Implementation for NS 2, Obtain via http://www4.ncsu.edu/ rhee/export/bitcp/bitcp-ns/bitcp-ns.htm

[26] S. Floyd, Limited Slow-Start for TCP with Large Congestion Windows, *RFC 3742*, March 2004.