Ann. Telecommun. (2010) 65:73–85
DOI 10.1007/s12243-009-0145-7

On the fairness characteristics of FAST TCP

Tuan A. Trinh · Balázs Sonkoly · Sándor Molnár

Received: 15 November 2008 / Accepted: 24 November 2009 / Published online: 15 December 2009

Abstract Fairness of competing transmission control protocol (TCP) flows is an integral and indispensable part of transport protocol design for next-generation, high-bandwidth-delay product networks. It is not just a protocol-intrinsic property but it could also have severe impact on quality of experience (QoE). In this paper, we revisit FAST TCP fairness behavior based on a comprehensive performance evaluation study. We demonstrate that FAST TCP with proper parameter settings can always achieve fair behavior with HighSpeed TCP and Scalable TCP. We also show that this behavior is a rather robust property of the protocol concerning different traffic mix or network topology. The dynamic behavior of reaching the fair equilibrium state can be different, which is demonstrated in the paper. Our study also emphasizes the important need for finding a dynamic sensitive fairness metric for performance evaluation of transport protocols for next-generation, high-bandwidth-delay product networks.

Keywords High-speed networks · FAST TCP · Fairness analysis

T. A. Trinh · B. Sonkoly (✉) · S. Molnár
High Speed Networks Laboratory,
Department of Telecommunications and Media Informatics,
Budapest University of Technology and Economics,
Magyar tudósok körútja 2, 1117, Budapest, Hungary
e-mail: sonkoly@tmit.bme.hu

T. A. Trinh
e-mail: trinh@tmit.bme.hu

S. Molnár
e-mail: molnar@tmit.bme.hu

1 Introduction

The advance in technologies and newer and newer forms of applications ensure improved and diversified services to end-users but they also bring new challenges for network designers. 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 [16, 30], and a number of solutions have been proposed (HighSpeed transmission control protocol (TCP) [8], Scalable TCP [12], FAST TCP [11, 28], and others). Roughly, these protocols can be divided into two classes: loss-based and delay-based. Loss-based versions share similar features with traditional TCP (TCP Reno), whereas delay-based TCP (FAST TCP) is an extension of TCP Vegas [15]. There is considerable research regarding the modeling and analysis of high-speed TCP versions, e.g., [8, 11, 12, 17, 22, 26]. It is widely accepted that one of the most important issues with these protocols is operability and deployability. This directly leads to the question of fairness. In fact, this question has been tackled by the research community for quite a long time (see, e.g., [1, 2, 11–13, 23, 24, 27, 29]), 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 are all concerned 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.

Fairness could also have significant impact on the subjective experience of the end users. For example, when more users download files (e.g., DVD content) from a file server sharing a common bottleneck link, may experience very different download times; in extreme cases, a user's transaction can last several times longer than others'. Therefore, investigating fairness characteristics, the aspects of quality of experience (QoE)—which is a "measure" from the user perspective—has to be considered, as well. User behavior—selfish behavior in particular—can have significant impacts on the performance of the network as a whole [26]. Selfish activities (e.g., by adopting aggressive transport protocols) of *some* users can be considered *unfair* to other users sharing the same network. In this respect, the fairness of network bandwidth allocation can implicitly affect the experience of end-users who suffer unfair bandwidth share. There is a huge diversity in actual network applications with versatile traffic characteristics having different impacts on the TCP fairness. As a consequence, it has been recognized that TCP is no longer sufficient to ensure the fair sharing of network resources. This is the reason why priority control may be introduced to support fairness. Moreover, there is an increasing tendency to employ application level control to improve users' performance over shared wide-area networks. For example, some web and peer-to-peer applications use Parallel TCP to accelerate data access and provide acceptable fairness [10, 14].

The new challenges of TCP have been addressed by several research groups in the last decade, and a number of new TCP versions have been developed. A brief overview of the protocols analyzed in the paper is given here, and some technical details are summarized in Table 1. HighSpeed TCP (HSTCP) [8] is a modification to TCP's congestion control mechanism for use with TCP connections with large congestion windows. It changes the TCP response function to achieve better performance on high-capacity links. HSTCP is based on an additive increase multiplicative decrease (AIMD) mechanism, where the increase and decrease parameters ($a(W)$ and $b(W)$) are functions of the current value of the congestion window (see the corresponding row of Table 1), yielding an adaptive and more or less scalable algorithm.

Ideas to introduce multiplicative increase multiplicative decrease (MIMD) mechanisms for TCP have also been considered. Scalable TCP (STCP) [12] is a good example that has been suggested as an efficient transport protocol for high-speed networks. Here, the multiplicative increase and multiplicative decrease algorithm guarantees the scalability of the protocol. The congestion window is increased by a constant parameter (a) as a response to a received acknowledgement, while it is reduced in a multiplicative manner (by bW) in case of packet losses (see Table 1).

The research on the delay-based ideas has resulted in FAST TCP [11, 28]. FAST TCP has the same equilibrium properties as TCP Vegas [15], but it can also achieve weighted proportional fairness. FAST TCP seeks to restrict the number of its packets queued through the network path between an upper (β) and a lower (α) bound; however, the behavior is usually controlled by a single parameter (α) that can be considered as the targeted backlog (packets in the buffers) along the flow's path [11, 17, 28]. Under normal network conditions, FAST TCP periodically updates its congestion window based on the comparison between the measured average round-trip time (RTT) and the estimated round-trip propagation delay (when there is no queueing). More exactly, the window is adjusted according to the formula presented in Table 1, where γ is the step size affecting the responsiveness of the protocol, and `baseRTT` is the minimum RTT observed so far, which is an estimation of the round-trip propagation delay. The parameter α controls the equilibrium behavior; therefore, the appropriate setting of this parameter is crucial (see [11, 17, 28]). FAST TCP also reacts to packet losses, halving its congestion window.

Due to its beneficial properties, the fairness characteristics of the delay-based scheme, especially FAST TCP, are intensively investigated by the research community. In [2], the equilibrium fairness properties of FAST TCP and TCP Vegas are compared, and the impacts of the error of the propagation delay estimation are analyzed. In [24], the intra-protocol fairness characteristics of FAST TCP and TCP Reno are investigated considering the long-term behavior and equilibrium bandwidth allocations. A novel metric is

Table 1 Details of TCP variants analyzed in the paper

Protocol	Window adjustment	(When)	Reaction to loss
HSTCP	$\mathtt{w} \leftarrow \mathtt{w} + \frac{a(\mathtt{w})}{\mathtt{w}}$	(Per ACK)	$\mathtt{w} \leftarrow \mathtt{w} - b(\mathtt{w})\mathtt{w}$
STCP	$\mathtt{w} \leftarrow \mathtt{w} + a$	(Per ACK)	$\mathtt{w} \leftarrow \mathtt{w} - b\mathtt{w}$
FAST TCP	$\mathtt{w} \leftarrow \min\left\{2\mathtt{w},\ (1-\gamma)\mathtt{w} + \gamma\left(\frac{\mathtt{baseRTT}}{\mathtt{RTT}}\mathtt{w} + \alpha\right)\right\}$	(Periodically)	$\mathtt{w} \leftarrow 0.5\mathtt{w}$

also introduced as the Euclidean distance between the max–min fair bandwidth allocation (as the benchmark) and any bandwidth allocation. On the basis of this framework, the fairness of FAST TCP and TCP Reno is evaluated, and numerical demonstrations are also given regarding different topologies, such as dumb-bell, parking-lot, and the NSFNET Backbone topologies. These works consider the long-term, average behavior of the protocols, and the dynamical, transient properties are not analyzed. Fairness problems of FAST TCP due to the inaccurate estimation of the propagation delay are revealed in [25], and a possible solution is also given. The fairness is improved by giving the first packet in every flow priority. Similar phenomena are investigated in [6], and a novel method is suggested in order to ameliorate the propagation delay estimation. This scheme does not need priority queueing; instead, the bottleneck queue is emptied occasionally by a pause of newly started flows. This algorithm can improve the fairness of FAST TCP to new coming flows.

In this paper, we revisit FAST TCP, a delay-based TCP version that is designed as a transport protocol for next-generation networks, especially high-bandwidth-delay product networks. We illustrate and show some surprising benefits of this approach, in particular, FAST TCP, in terms of fairness. As the intra-protocol characteristics of the delay-based protocols, such as TCP Vegas and FAST TCP, are well investigated by the research community (see, e.g., [2, 6, 17, 24–26]), this paper deals with open issues regarding the interaction of delay-based and loss-based schemes. Our study also emphasizes the important need for finding a dynamic sensitive fairness metric for performance evaluation of high-speed transport protocols. As a first step towards this new fairness metric, we introduce the *saturation time* of transport protocols and suggest novel methods—such as spectral analysis and identification of *main operating frequencies*—that can play an important role in the characterization of new proposals. With this spectral analysis together with flow-level, packet-level, and queueing analysis, we have a good understanding of the investigated phenomena. This paper is devoted to giving important and demonstrative results gained from a comprehensive analysis [21].

The rest of the paper is organized as follows: In Section 2, the simulation environment and the important parameters are presented. Section 3 shows the fairness issues of FAST TCP with other loss-based TCP versions from a flow-level perspective. Observations of the deficiencies of the available fairness metrics are also provided in Section 3. Section 4 provides a comprehensive packet-level analysis of the observed phenomena, especially the impact of starting time on long-term fairness. In Section 5, a brief discussion on the performance in more complex topologies is given. Section 6 concludes the paper.

2 Simulation environment

The fairness analysis of competing high-speed TCP protocols and the validation of the analytical results are carried out in the Ns-2 [18] simulation environment. Our simulation scripts regarding different network scenarios can be found in [21]. The different high-speed transport protocols are integrated in the environment. Ns-2 version 2.27 includes the algorithm of HSTCP, while the STCP control mechanism can easily be implemented. The Ns-2 source code of FAST TCP is used from [19] simulator module for ns-2.

The examined dumb-bell topology containing one bottleneck link is shown in Fig. 1a. The queueing mechanism corresponding to the bottleneck link is drop-tail. We do not consider the impacts of the buffer size (B) in our analysis, and the 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 (Fig. 1b), 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 (permanent TCP connection). 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., with a 50-s delay), and on the other hand, scenarios with smaller delay (e.g., with a 15-s delay) are also examined. In the dumb-bell topology, the competition of a later-entering flow against a traffic aggregate

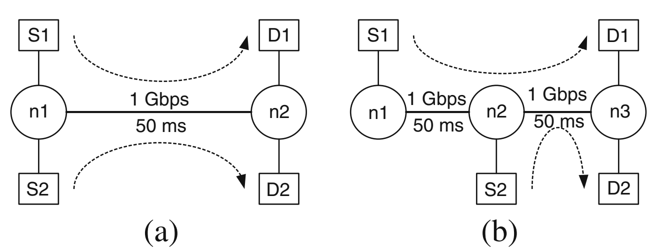


Fig. 1 Network topologies. **a** Dumb-bell. **b** Simple parking-lot

Table 2 Parameters

Network parameters	
Capacity	1 Gbps
RTT	100 ms
Packet size	1,500 bytes
Buffer size (B)	
Dumb-bell	8,333 pkts
Parking-lot	25,000 pkts
Sampling periods	
`cwnd`, queue	0.01 s
Throughput	1 s
FAST TCP—dumb-bell	
$\alpha = \beta$	4,166 pkts
FAST TCP—parking-lot	
$\alpha = \beta$	12,500 pkts

containing ten flows using the same protocol is also analyzed.

Besides permanent (infinite) connections, we also investigate the interaction of finite FTP transactions downloading a constant number of packets (bytes). In these cases, the transaction times are directly experienced by end users and mainly determine the quality of experience.

During the evaluation, the default parameter set of the protocols is used (see [8] and [12]). HSTCP and STCP apply the limited slow-start (LSS) mechanism [9], as well. The parameters of the simulations are summarized in Table 2.

FAST TCP seeks to restrict the number of its packets queued through the network path between an upper and a lower bound. The appropriate setting of parameters α and β regarding the bounds is crucial. We use only one parameter (α) setting the bounds as $\alpha = \beta$. The control mechanism is based on the comparison between the observed RTT and the `baseRTT`, which is an approximation of the round-trip propagation delay (when there is no queueing). The α parameter of FAST TCP flows is chosen so that the total number of outgoing packets of the flows is smaller than the buffer size (B) to avoid losses due to buffer overflow for these FAST TCP connections. In case of two flows, then the α parameter of FAST TCP is set to $B/2$.

3 Flow-level study

In this section, two competing flows (one FAST TCP flow and one HSTCP or STCP flow) are examined at the flow-level and the dynamics of average throughput and fairness metrics are analyzed. The deficiencies of the available fairness metrics are also revealed. Recent researches [21, 29] showed that competing high-speed transport protocols (such as STCP, HSTCP) cannot always achieve fair equilibrium state or the convergence time can be too long. It was also revealed that the starting time of the flows can have relevant impact on the performance. We show that FAST TCP with appropriate parameters can exhibit fair or almost fair behavior. Here, we focus on the inter-protocol properties; however, FAST TCP with appropriate parameters shows fair behavior with other FAST TCP flows, as well.

First, the dumb-bell topology is analyzed. The dynamics of the bandwidth share for scenarios with different starting delays are presented in Fig. 2. It can be observed in all presented results that the equilibrium bandwidth share is approximately fair. On the one hand, when FAST TCP starts first, the bottleneck bandwidth is always shared fair in equilibrium state. On the other hand, when FAST TCP enters the network later, the equilibrium state is only near to the fair state. In the rest of the paper, this bandwidth share is referred to as almost fair. This operation is reached after a transient period with a length depending on the starting delay. This bias in the equilibrium state can be caused by the estimation error of `baseRTT` (the FAST TCP flow does not experience empty buffer).

For the examined scenarios, available fairness metrics can also be derived. Here, we calculate three important ones that can be used in our analysis. Let $\overline{\mathrm{BW}}_1$ and $\overline{\mathrm{BW}}_2$ be the average bandwidth share of the two sources, respectively.

Relative fairness (used in [8]) can be defined as follows:

$$\mathrm{RF} = \frac{\overline{\mathrm{BW}}_1}{\overline{\mathrm{BW}}_2}.$$

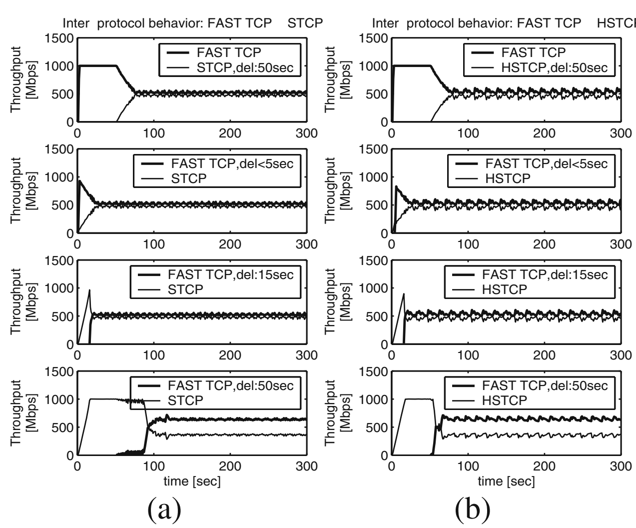


Fig. 2 Performance of competing flows. **a** FAST TCP–STCP. **b** FAST TCP–HSTCP

Jain's index [5] is a normalized metric in the [0.5, 1] interval and can be defined as follows:

$$\mathrm{JI} = \frac{(\overline{\mathrm{BW}}_1 + \overline{\mathrm{BW}}_2)^2}{2(\overline{\mathrm{BW}}_1^2 + \overline{\mathrm{BW}}_2^2)}$$

The bandwidth share is fair if $\mathrm{JI} \to 1$, and unfair behavior can be observed if $\mathrm{JI} \to 0.5$.

Another normalized asymmetry metric proposed in [4] can express which flow is more aggressive. This index is defined as

$$\mathrm{BI} = \frac{\overline{\mathrm{BW}}_1 - \overline{\mathrm{BW}}_2}{\overline{\mathrm{BW}}_1 + \overline{\mathrm{BW}}_2}.$$

The closer BI is to 0, the more fair bandwidth share can be observed. $\mathrm{BI} \to 1$ shows the dominance of the first flow while $\mathrm{BI} \to -1$ shows the dominance of the second one.

Table 3 summarizes the fairness indices of the protocols calculated from the simulations. The results confirm quantitatively the qualitative statements based on throughput diagrams. These metrics are mainly capable of characterizing the equilibrium behavior and cannot capture the dynamic properties of the interaction; thus, further analysis both in packet-level and flow-level is necessary.

Second, we examine the behavior of FAST TCP in the parking-lot topology where the fairness of competing high-speed TCP flows with different RTT can be analyzed. The results are promising, and here, some examples are given as illustrations. We found that FAST TCP with longer path and RTT can also achieve good performance and shows fair behavior with STCP and HSTCP. Another attractive property of the protocol is shown when the RTT of the FAST TCP flow is shorter. FAST TCP does not starve the other flows and fair equilibrium states are achieved. Demonstrative results are shown in Fig. 3. The upper parts of the figure correspond to scenarios when the RTT of the FAST TCP flow is greater, while the lower parts present the results for the reverse case. When FAST TCP flow starts first, the characteristics of the interaction are similar to the behavior exhibited in the dumb-bell topology: after a transient phase, FAST TCP gives up approximately half of the shared link bandwidth. However, the length of the transient period differs. In case of later-entering FAST TCP flow, the fair (or almost fair) equilibrium state is realized again; however, in the transient phase, other phenomena can also be observed. The later-entering FAST TCP flow with longer RTT can only achieve the fair equilibrium at the cost of timeout of the other flow. In our simulation environment, HSTCP and STCP can respond to this event in a different manner; thus, the transient phases show different properties.

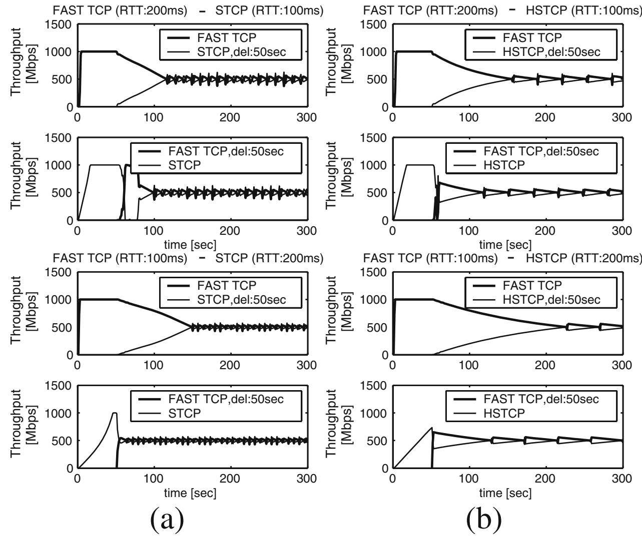


Fig. 3 Performance—parking-lot topology. **a** FAST TCP–STCP. **b** FAST TCP–HSTCP

Finally, we show some illustrative results on the interaction of finite TCP flows and the impacts of flow size on the fairness characteristics. In [7], it is argued that flow completion time (FCT) is one of the most important performance metrics for the user and can be defined as the time from when the first packet of the connection is sent (typically a SYN packet) till when the last packet is received. From the aspects of high-speed transport protocols, larger file sizes (e.g., CD or DVD downloads) are more interesting because the new congestion control algorithms take effect after the initial slow-start phase. However, for example, web applications—generating a lot of short-time connections (dragonflies, mice traffic)—have a great importance today [3]. Therefore, the number of small

Table 3 Fairness indices (dumb-bell topology)

Flow 1	Flow 2	del: 50s			del: 15s			del < 5s		
Prot.	Prot.	BI	JI	RF	BI	JI	RF	BI	JI	RF
FAST TCP	STCP	0.051	0.997	1.106	0.051	0.997	1.107	0.051	0.997	1.107
STCP	FAST TCP	−0.273	0.930	0.569	−0.051	0.997	0.904	−0.051	0.997	0.905
FAST TCP	HSTCP	0.081	0.993	1.177	0.081	0.994	1.176	0.081	0.994	1.176
HSTCP	FAST TCP	−0.287	0.924	0.555	−0.081	0.994	0.851	−0.081	0.993	0.848

objects (files) on web pages that do not allow TCP to go beyond the slow-start phase can have a significant impact on the download times of web pages. For example, in [20], traffic mixes containing a great number of short-lived and long-lived flows are analyzed, and it is shown that the good RTT-fairness properties of delay-based control do not hold due to the slow-start behavior of short flows. However, we focus on the interaction of the high-speed control algorithms, and the impacts of short-lived flows operating in only slow start phase are not analyzed here.

FCT—as a metric—can be convenient for characterizing the competition of finite transport protocol flows (e.g., performing finite FTP downloads), as well. For competing TCP flows, the FCT consists of two main components. On the one hand, FCT is obviously affected by the sending rate control mechanism of the protocol (this part captures the individual flow behavior). And on the other hand, the interaction/competition of the flows results in additional time necessary to accomplish the transaction (this component regards to the fairness characteristics of the protocols). The separation of these effects could be practical. We define reference FCT values for the examined protocols as the FCT for a given flow size when a single flow of that protocol operates alone in the same network environment. In the scenarios with competing flows, the FCT values can be related to the reference values of the protocols, respectively. In Fig. 4a, the relative FCT (the ratio of FCT and the reference FCT of the protocol) are shown for different flow sizes when a FAST TCP flow competes with a STCP flow in the dumb-bell topology (see Fig. 1a). The parameters of the simulations are shown in Table 2. Here, the flows always start at the same time instant and download the same amount of data while the packet size is constant (1,500 bytes). We emphasize that this plot corresponds to several scenarios with different flow sizes. It can be observed that the FCT of STCP and FAST TCP increase together from the value of 1 toward 2 in the presented range. Therefore, a similar behavior is experienced by both users. It is worth noting that this beneficial property is not always shown by other protocols. As an example, the relative FCT of competing STCP and HSTCP flows are presented in Fig. 4b. For the examined range of flow sizes, the user of the STCP protocol experiences similar FCTs to the reference case (when he/she does not share the network with others), while the other user (choosing HSTCP) meets sometimes more than two times longer completion times than the ideal ones.

For the sake of completeness, the FCT are presented for scenarios when the flow sizes are significantly smaller than previously. The corresponding plots are shown in Fig. 5a and b for competing STCP–FAST TCP and STCP–HSTCP flows, respectively. Here, the horizontal axes follow logarithmic scale. The beneficial fairness properties of FAST TCP are not exhibited in these scenarios and the relative FCT values indicate that FAST TCP slightly outperforms the STCP flows when the flow size is smaller. (See the smaller values of

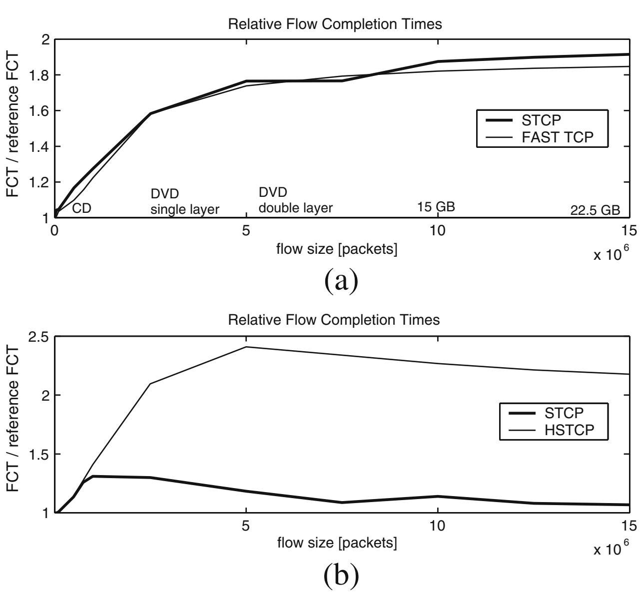


Fig. 4 Relative FCT, large flow sizes. **a** STCP–FAST TCP. **b** STCP–HSTCP

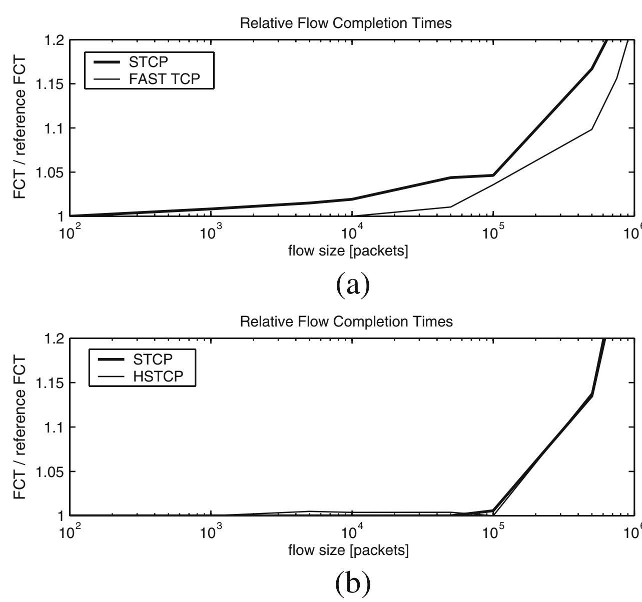


Fig. 5 Relative FCT, smaller flow sizes. **a** STCP–FAST TCP. **b** STCP–HSTCP

relative FCT in Fig. 5a.) On the other hand, HSTCP and STCP flows experience approximately the same completion times in these scenarios.

4 Packet-level analysis

This section is devoted to revealing and explaining the phenomena experienced at the flow-level based on a comprehensive packet-level analysis. First, the initial behavior of individual flows is summarized in order to gain a basic knowledge of the characteristics of different congestion control principles. Second, the long-term behavior of competing high-speed TCP protocols is investigated. We apply spectral analysis of `cwnd` and queue processes to get an insight into the dynamic properties of the interaction.

4.1 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. Here, the investigation is carried out considering the simple dumb-bell topology. 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 Fig. 6a, the saturation time and different phases of an individual STCP 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. For a delay-based protocol, depending on the network environment (buffer size, parameters of the protocol), packet losses can be avoided during the operation. In these cases, the network is said to be saturated when the congestion window has settled down around the equilibrium state or the source has entered the delay-based (AIAD) operating regime (see Fig. 6c).

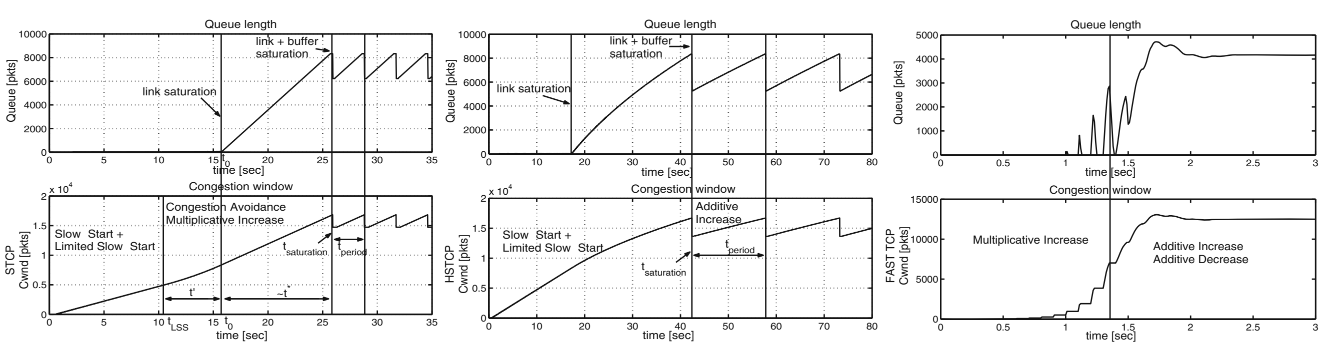
Various TCP versions apply different mechanisms during the initial phase. A source generally starts sending according to a slow-start-like manner using a multiplicative increase algorithm with a protocol-dependent parameter. This means that the congestion window is increased by a constant value for each acknowledgement received. In our particular cases, the protocols use the following mechanisms. The behavior of HSTCP and STCP is determined by the slow-start and LSS algorithms. With certain network parameters, the LSS phase can be left for the additive increase (HSTCP) or the multiplicative increase (STCP) phase, before the first packet drop. FAST TCP increases its congestion window according to a multiplicative increase algorithm if it is far from the equilibrium state. As a result, to understand the saturation behavior of different protocols, we have to understand the operation of basic algorithms used in the initial phases.

4.1.1 Dynamics of slow-start

The slow-start mechanism is analytically tractable and relevant parameters can easily be derived. Here, we only summarize the main results (for further details, see [21]).

The slow-start phase lasts till `cwnd` reaches the threshold `ssthresh` or a packet loss can be detected. This time instance is $t_{ss} = \min\{t_{th}, t_{\text{drop}} + t_{\text{delay}}\}$, where $t_{th} = R_0 \log_2 \text{ssthresh}$ and t_{drop} is the solution of the following equation:

$$\frac{2^{t/R_0}}{\lg 2} - Ct + \left(Ct_0 - \frac{2^{t_0/R_0}}{\lg 2} - B\right) = 0, \quad (1)$$



(a) (b) (c)

Fig. 6 Saturation time and equilibrium behavior. **a** STCP. **b** HSTCP. **c** FAST TCP where R_0 is the round-trip propagation delay, C is the link capacity, and B is the buffer size. The time instance t_{drop} can be approximated by the following simple formula, assuming that the congestion window is equal to the sum of the bandwidth-delay product (BDP) and the buffer size at the saturation time:

$$t_{\text{drop}} \approx R_0 \log_2(R_0 C + B) = R_0 \frac{\lg(R_0 C + B)}{\lg 2}. \quad (2)$$

t_{delay} corresponds to queueing delay and one-way propagation delay.

Configuring initial slow-start threshold to be 100 packets, the end of the slow-start phase is triggered by the event that `cwnd` exceeds `ssthresh`. During our analysis, we always assume this initial value of `ssthresh`.

4.1.2 Dynamics of LSS

LSS operates in congestion avoidance mode in the Ns-2 implementation till the first packet drop. LSS affects the increase mechanism of `cwnd`, comparing the increment of the congestion control mechanism (e.g., STCP, HSTCP) with its own increment, and the maximum of these values is used. With this algorithm, a faster convergence can be achieved when the source's sending rate is far from the equilibrium value. In LSS phase, `cwnd` is increased by at most `max_ssthresh`/2 per RTT. In the simulations, the parameter is set to the proposed value (100 packets). It can easily be derived analytically [9] that

$$\log_2(\texttt{max\_ssth}) + \frac{\texttt{cwnd} - \texttt{max\_ssth}}{\texttt{max\_ssth}/2}$$

RTT is needed to reach `cwnd` (which is greater than the threshold parameter). The first term corresponds to the standard slow-start phase and max_ssth stands for max_ssthresh.

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 protocol version. As `cwnd` increases, there will be a state (W_{LSS}) when the increment of LSS and the increment of STCP's or HSTCP's algorithm will be equal, triggering the end of this phase. On the one hand, in case of individual STCP flow, this state can be expressed by

$$W_{\text{LSS}} = \frac{\texttt{max\_ssth}}{2} \frac{1}{a},$$

where a is the increase parameter of STCP. The details of the derivation can be found in [21]. Our parameters give that the end of LSS phase is expected to be around $W_{\text{LSS}} = 5{,}000$ at $t \approx 10.46$ s. On the other hand, for HSTCP flow, W_{LSS} can be derived from the following equation:

$$a(W_0) = \texttt{max\_ssth}/2,$$

where $a(W_0)$ is the `cwnd`-dependent increase parameter of HSTCP. In our scenario, $W_{\text{LSS}} \approx 29{,}000$. This high value of `cwnd` cannot be reached with the computed simulation parameters, resulting in HSTCP source operating in LSS till the first packet drop. Thus, the initial behavior is determined by slow-start and LSS algorithms.

4.1.3 STCP—saturation time

In case of STCP, the end of LSS phase can be expressed as follows (see [21] for details):

$$t_{\text{LSS}} = R_0 \frac{\lg \texttt{max\_ssth}}{\lg 2} + R_0 \frac{W_{\text{LSS}} - \texttt{max\_ssth}}{\texttt{max\_ssth}/2} \approx 10.46\,\text{s}, \quad (3)$$

where W_{LSS} is the value of congestion window triggering the end of LSS. After LSS, the multiplicative increase mechanism of the protocol operates. During this period, the congestion window is increased from W_{LSS} to the BDP ($R_0 C$). Thus, the link saturation time can easily be determined:

$$t_0 = t_{\text{LSS}} + t' = t_{\text{LSS}} + R_0 \frac{\lg \frac{R_0 C}{W_{\text{LSS}}}}{\lg(1+a)} \approx 15.6\,\text{s}. \quad (4)$$

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 $W_0 = R_0 C$ to $R_0 C + B$ according to the multiplicative increase mechanism. Approximating the increase of the queueing delay by a linear function, the RTT 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:

$$\hat{t}_{\text{saturation}} = t_0 + t^* = t_0 + \tilde{R} \frac{\lg \frac{R_0 C + B}{R_0 C}}{\lg(1+a)} \approx 26.05\,\text{s}. \quad (5)$$

The analytically derived parameters and the approximation of saturation time meet well the simulation results presented in Fig. 6a.

4.1.4 HSTCP—saturation time

In case of individual HSTCP flow, the time of the first packet drop (saturation time) can similarly be determined as it was outlined for STCP. The link saturation time can be expressed as follows:

$$t_0 = R_0 \frac{\lg \mathtt{max\_ssth}}{\lg 2} + R_0 \frac{R_0 C - \mathtt{max\_ssth}}{\mathtt{max\_ssth}/2} \approx 17.13\,\text{s}. \quad (6)$$

Determining the saturation time, a similar approximation can be used as it was applied for STCP:

$$\hat{t}_{\text{saturation}} = t_0 + t^*,$$

where

$$t^* = \tilde{R} \frac{R_0 C + B - R_0 C}{\mathtt{max\_ssth}/2} = \tilde{R} \frac{B}{\mathtt{max\_ssth}/2}. \quad (7)$$

Our parameters yield that $\hat{t}_{\text{saturation}} = 42.13$ s which meets well the simulation results (see Fig. 6b).

4.1.5 FAST TCP—saturation time

Far from the equilibrium state of `cwnd`, FAST TCP converges exponentially to that equilibrium performing a slow-start-like multiplicative increase algorithm. This fast convergence of the `cwnd` is shown in Fig. 6c. If queueing delay exceeds a threshold (which is a constant parameter of the protocol in the Ns-2 implementation), the additive increase additive decrease control algorithm is used instead of multiplicative increase. In our simulation environment, FAST TCP (with $\alpha = 4166$) reaches the equilibrium approximately after 2 s.

4.2 Equilibrium behavior

In this section, the packet-level characteristics of the long-term behavior of competing high-speed TCP flows are investigated.

4.2.1 Dumb-bell topology: single flows

First, we focus on scenarios when FAST TCP source starts the transmission and the other flow enters into the network when the first one has achieved maximal sending rate. The simulation results corresponding to starting delay of 50 s are presented in Fig. 7 for STCP (a) and HSTCP (b), respectively. During the transient phase, the additive decrease algorithm of FAST TCP interacts with the control mechanism of the other protocol (LSS and multiplicative increase in case of STCP and LSS in case of HSTCP). Following the proposed schemes for choosing an α parameter of FAST

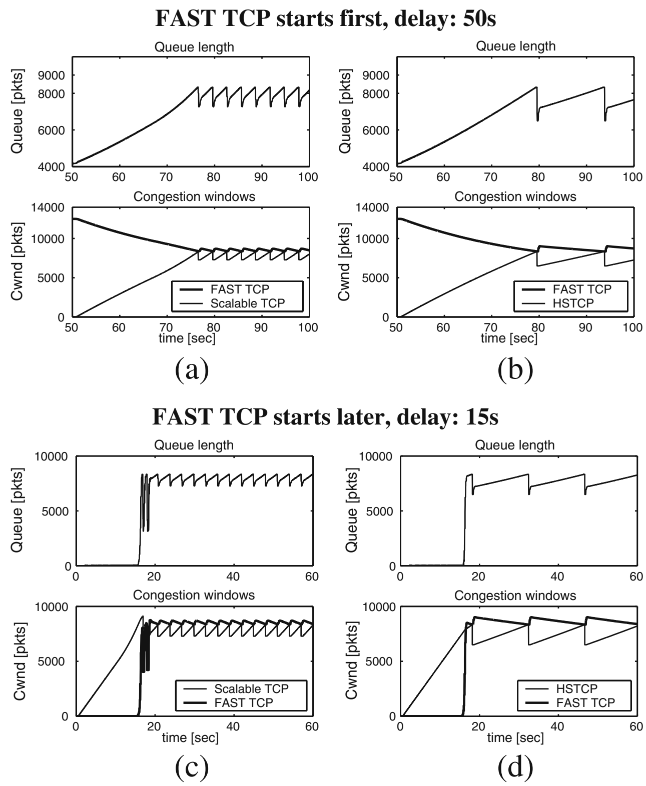


Fig. 7 Competition of two flows. **a** FAST TCP–STCP. **b** FAST TCP–HSTCP. **c** STCP–FAST TCP. **d** HSTCP–FAST TCP

TCP, `cwnd` processes converge to an equilibrium state corresponding to approximately fair bandwidth share. The length of the transient phase is determined by the congestion control algorithms. After the transient period, a common periodic behavior is shown by the sources. When STCP reduces its congestion window, FAST TCP can increase the number of packets in the bottleneck queue performing an additive increase based on queueing delay. During the second part of a period, the multiplicative increase of STCP interacts with the additive decrease of FAST TCP. Thus, the periodic behavior is affected by the interaction of AIAD–MIMD algorithms. The common time period and the dynamics of the bottleneck queue are determined by the time period of STCP. It is worth noting that losses do not occur during the FAST TCP connection and the equilibrium state is quasistable. The equilibrium behavior of FAST TCP and HSTCP is very similar. Here, the interaction of AIAD and AIMD mechanisms results in a longer time period.

Second, the FAST TCP source enters later into the network and tries to catch the half of the capacity of the bottleneck link. Recent researches [21, 29] showed that a STCP flow in equilibrium state can starve other flows starting later (including other STCP flows). FAST TCP with α parameter chosen as suggested in [11] achieve significant bandwidth share against STCP and HSTCP, too. The simulation results corresponding to 15 s delay are presented in Fig. 7c and d. In these scenarios, after a very short transient period, the congestion windows settle down again around an equilibrium state.

A significantly different behavior can be experienced at the packet-level increasing the starting delay of the FAST TCP flow. As an illustration, the simulation results of the competition of STCP and FAST TCP flow corresponding to a 50-s delay are shown in Fig. 8. This behavior can be examined in the frequency domain, too. The power spectral density (PSD) functions of the `cwnd` process of STCP and FAST TCP and the bottleneck queue process are also shown in Fig. 8. The good performance of FAST TCP can be explained by the special control algorithm used by the protocol. When FAST TCP is far from the equilibrium sending rate, it performs multiplicative increase algorithm. As the bottleneck queue operates around its full state, during the transient period, FAST TCP also suffers from losses and halves the `cwnd`. After a recovery period, the exponential increase is performed until the next loss. After a long and oscillating transient phase, the previously seen common periodic equilibrium behavior is exhibited when FAST TCP does not suffer from losses. The dominant frequency of a single STCP flow ($\omega \approx 0.34\,1/\mathrm{s}$) occurs in the PSD of FAST TCP (with lower energy value), as well, while the presence of a higher frequency component can also be observed corresponding to the MIMD oscillation of the transient phase. These two frequency spikes mainly determine the dynamics of the bottleneck queue. In case of interaction with HSTCP, a similar behavior can be observed (see Fig. 9). We observed that the length of the transient period depends on the starting time and other parameters of FAST TCP, as well.

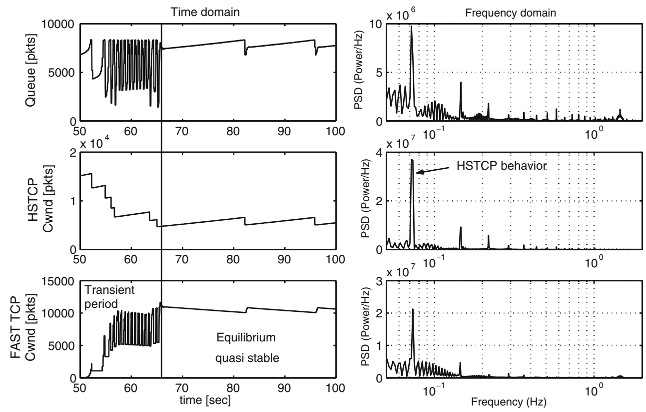


Fig. 9 HSTCP–FAST TCP, delay: 50 s

4.2.2 Dumb-bell topology: traffic aggregate—single flow

FAST TCP can also achieve good performance and fair behavior against STCP or HSTCP traffic aggregate. On the one hand, we found that later entering FAST TCP flow can occupy half of the bottleneck bandwidth beside HSTCP flows (if the parameters are well chosen) and the behavior is similar to the behavior of two competing flows: FAST TCP realizes a quasistable equilibrium state without losses (AIAD). On the other hand, the interaction with STCP traffic aggregate can have different characteristics. To illustrate the point, we let a traffic aggregate that contains ten STCP flows (Fig. 10) and ten HSTCP flows (Fig. 11) mixed with a FAST TCP source enter with a delay of 50 s. Here, the characteristics of the equilibrium behavior are determined by the MIMD mechanisms of the protocols, and FAST TCP is not able to reach a stable (or quasistable) state and shows oscillation. However, the achieved throughput approximates the half of the bottleneck capacity (as it

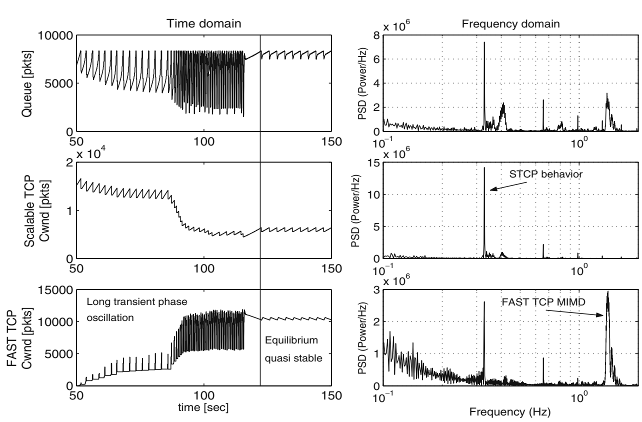


Fig. 8 STCP–FAST TCP, delay: 50 s

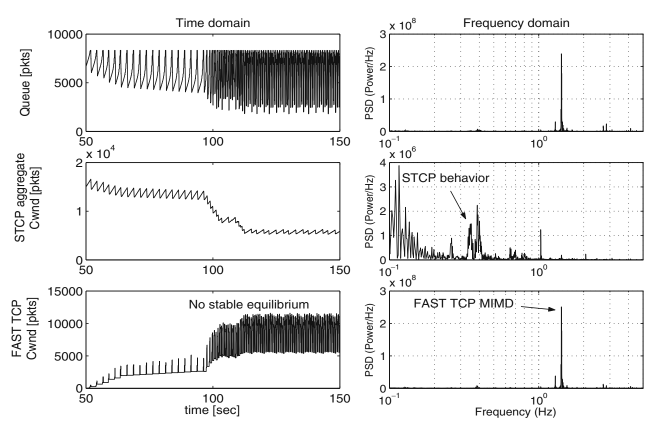


Fig. 10 STCP aggregate—FAST TCP

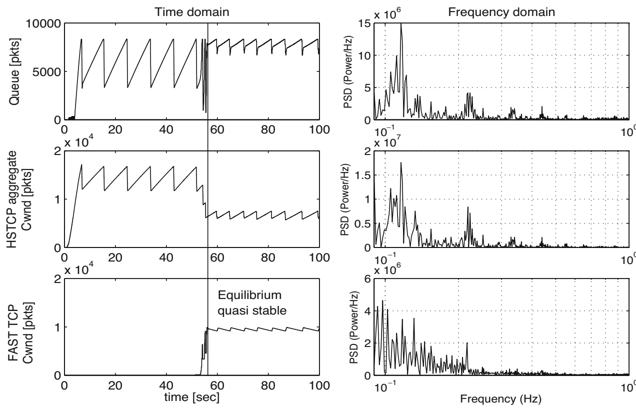


Fig. 11 HSTCP aggregate—FAST TCP

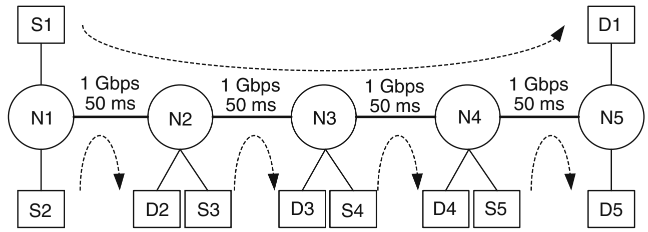


Fig. 13 Complex parking-lot topology with four congested links

is targeted by the parameter setting of FAST TCP) resulting in good performance.

4.2.3 Parking-lot topology

We found that the fair behavior of FAST TCP still holds in simple parking-lot topology with one congested link where the flows meet different RTT. Here, we show some demonstrative results. (For further details, see [21].) In the first scenario, the FAST TCP flow traverses the shorter path. As an illustration, the simulation results are shown in Fig. 12a for later-entering STCP. Besides the dynamics of the bottleneck queue and the congestion windows, the figure includes the throughput diagrams, too. Similar to the competition in the dumb-bell topology, fair or almost fair behavior is always exhibited. The FAST TCP flow with shorter RTT does not starve the loss-based protocols with longer paths. FAST TCP reduces its sending rate till a fair equilibrium bandwidth share is realized. The only difference in the long-term behavior is the oscillation of the `cwnd` process of FAST TCP indicating packet losses. In the next scenario, FAST TCP flow operates with longer RTT. The illustrative results are presented in Fig. 12b. Here, FAST TCP enters into the network 50 s later than HSTCP. It can be observed that the FAST TCP flow can always achieve fair or almost fair equilibrium state. In certain cases, FAST TCP outperforms the loss-based protocol, which can be explained by the estimation error of `baseRTT` (FAST TCP flow does not experience empty buffer).

5 Discussion

Surprisingly, the performance of FAST TCP shows degradation in complex parking-lot topology with multiple congested links. Here, we show some demonstrative results (for further details, see [21]).

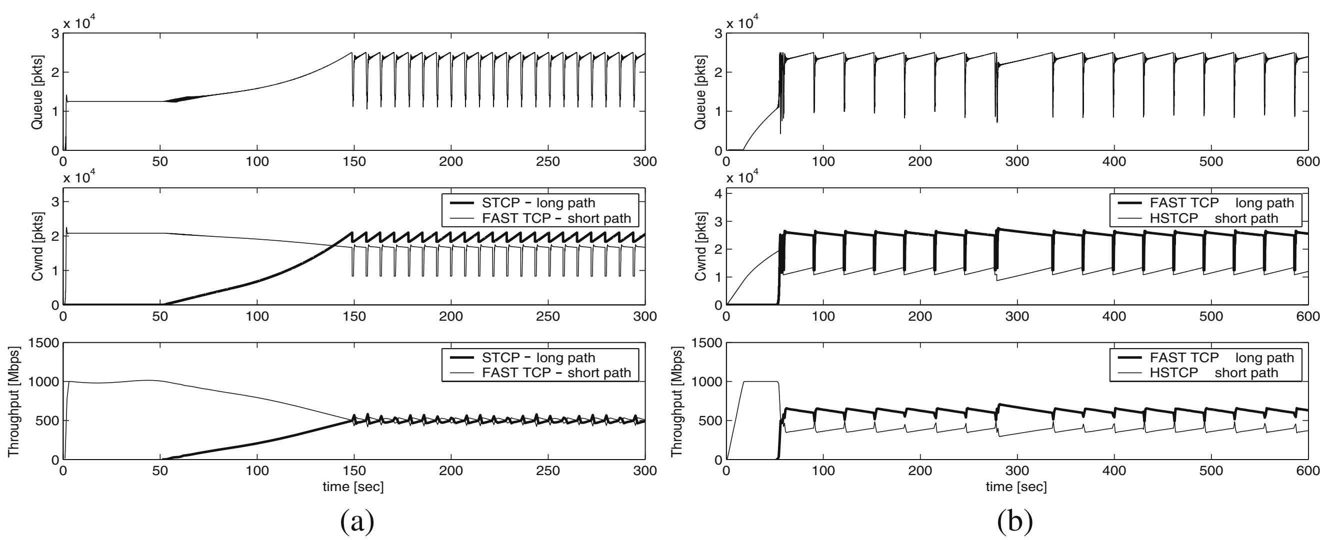


Fig. 12 Simple parking-lot topology: loss-based protocol—FAST TCP. **a** Scalable TCP on the longer path starts later. **b** HSTCP on the shorter path starts first

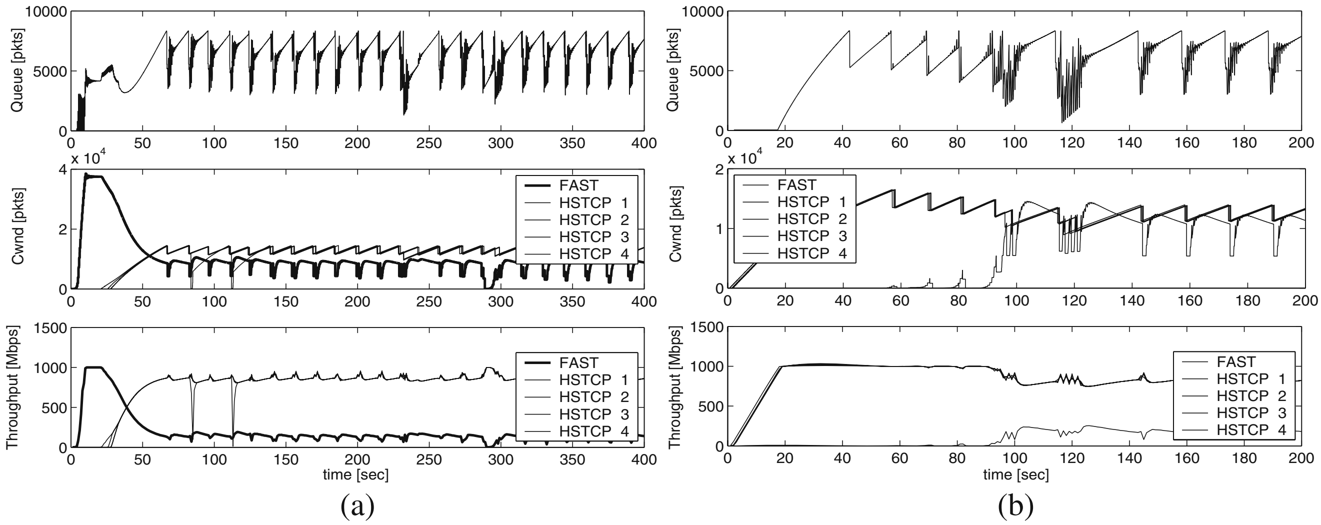


Fig. 14 Parking-lot: five nodes, FAST TCP–HSTCP. **a** FAST TCP with longer RTT starts first. **b** FAST TCP with longer RTT starts later

A complex parking-lot topology with five nodes is shown in Fig. 13, where all links are congested. Here, one FAST TCP flow traverses across the backbone containing four congested links and four flows of a loss-based protocol use single links, respectively. Here, the parameters of FAST TCP are set to the same value as were used in the dumb-bell scenarios.

As an illustration, the competition of a single FAST TCP flow traversing the backbone and four HSTCP flows using separate links is shown in Fig. 14. The congestion window of FAST TCP can settle down around the same equilibrium state where the other flows operate. Therefore, the bandwidth share of FAST TCP is significantly below the fair state. The equilibrium behavior slightly depends on the starting time of the flows, as well. Obviously, the performance of FAST TCP can be enhanced, increasing the α parameter of the protocol. However, this can yield unstable network behavior with degraded link utilization.

6 Conclusion

In this paper, we revisited FAST TCP, a delay-based TCP version proposed for next-generation networks. We illustrated and showed some surprising benefits of this approach in terms of fairness.

Our main findings are the following: *In contrast to loss-based protocols, FAST TCP with appropriate parameters can always show fair (or almost fair) behavior beside HSTCP and STCP flows in simple network environments with single congested link.* Concerning the dynamics of TCP starting times, the fair or almost fair state is achieved by different ways:

- If FAST TCP flow starts first, then a fair and quasistable equilibrium state can always be directly achieved.
- In case of a later entering FAST TCP flow, the equilibrium state is reached through an oscillating transient phase with a length depending on the starting time and other parameters.

These beneficial fairness properties implicitly affect the experience of end-users, as well. In contrast to aggressive protocols, such as STCP, *FAST TCP does not starve other flows, and it is able to significantly improve the quality of experience of different users.* More specifically, as the bandwidth is always shared fairly, the FCT experienced by end-users are very similar for a wide range of scenarios.

We have also found that this fair behavior of FAST TCP seems to be a robust property of the protocol and FAST TCP can achieve good utilization against traffic aggregate of loss-based protocols. However, in general, FAST TCP cannot exhibit these good properties operating in complex network environments (parking-lot topology) with multiple congested links. We should also note that this property holds for a certain range of the parameter alpha depending on the actual network topology, flow parameters, etc. To find a method that can continuously change this parameter according to the network and flow environments to keep this property broadly general is a good point of future research.

We have also revealed the drawbacks of currently used fairness metrics and showed the urgent need to find metrics which can reflect the dynamic protocol behavior. In the future, we plan to continue the analysis with other high speed protocols and more complex network environments. Our aim is to define a dynamics sensitive fairness metric for performance evaluation of transport protocols for next generation high BDP networks.

References

1. Altman E, Avrachenkov KE, Prabhu BJ (2005) Fairness in MIMD congestion control algorithms. In: Proceedings of IEEE Infocom 2005, vol 2, Miami, pp 1350–1361
2. Andrew LLH, Tan L, Cui T, Zukerman M (2005) Fairness comparison of FAST TCP and TCP Vegas. In: Proceedings of 19th international teletraffic congress, ITC19, Beijing
3. Brownlee N, Claffy K (2002) Understanding internet traffic streams: dragonflies and tortoises. IEEE Commun Mag 40(10):110–117
4. Bullot H, Cottrell RL, Hughes-Jones R (2003) Evaluation of advanced TCP stacks on fast long-distance production networks. Journal of Grid Computing 1(4):345–359
5. Chiu DM, Jain R (1989) Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. Comput Netw ISDN Syst 17(1):1–14
6. Cui T, Andrew LLH, Zukerman M, Tan L (2006) Improving the fairness of FAST TCP to new flows. IEEE Commun Lett 10(5):414–416
7. Dukkipati N, McKeown N (2006) Why flow-completion time is the right metric for congestion control. ACM SIGCOMM Comput Commun Rev 36(1):59–62
8. Floyd S (2003) Highspeed TCP for large congestion window. IETF RFC 3649
9. Floyd S (2004) Limited slow-start for TCP with large congestion windows. IETF RFC 3742
10. Hacker TJ, Noble BD, Athey BD (2004) Improving throughput and maintaining fairness using parallel TCP. In: Proceedings of IEEE Infocom 2004, vol 4, Hong Kong, pp 2480–2489
11. Jin C, Wei DX, Low SH (2004) FAST TCP: motivation, architecture, algorithms, performance. In: Proceedings of IEEE Infocom 2004, vol 4, Hong Kong, pp 2490–2501
12. Kelly T (2003) Scalable TCP: improving performance in highspeed wide area networks. ACM SIGCOMM Comput Commun Rev 33(2):83–91
13. Li Y, Leith D, Shorten RN (2007) Experimental evaluation of TCP protocols for high-speed networks. IEEE/ACM Trans Netw (ToN) 15(5):1109–1122
14. Liu Y, Gong W, Shenoy P (2001) On the impact of concurrent downloads. In: Proceedings of winter simulation conference, 2001, vol 2, Arlington, pp 1300–1305
15. Low SH, Peterson L, Wang L (2002) Understanding Vegas: a duality model. J ACM 49(2):207–235
16. Molnár S, Altman E (ed) (2005) Telecommunication systems, special issue on high speed transport protocols, vol 30, no 4
17. Molnár S, Kardos S, Trinh TA (2005) Parameter setting games in TCP Vegas and FAST TCP. In: Proceedings of 19th international teletraffic congress, ITC19, Beijing
18. Ns-2 Network Simulator. http://www.isi.edu/nsnam/ns
19. FAST TCP simulator module for ns-2. http://www.cubinlab.ee.unimelb.edu.au/ns2fasttcp/
20. Shimonishi H, Sanadidi M, Murase T (2007) Assessing interactions among legacy and high-speed TCP protocols. In: Proceedings of fifth international workshop on protocols for FAST long-distance networks (PFLDnet 2007). Marina Del Rey
21. Sonkoly B, Trinh TA, Molnár S (2007) Benchmarking high speed TCP fairness. Technical report, Budapest University of Technology and Economics, BME
22. Sonkoly B, Trinh TA, Molnár S (2005) Understanding High-Speed TCP: a control-theoretic perspective. In: Proceedings of the third IASTED international conference on communications and computer networks (CCN 2005), Marina del Rey
23. Souza E, Agarwal D (2003) A HighSpeed TCP study: characteristics and deployment issues. Technical report LBNL-53215, Lawrence Berkeley National Lab
24. Tan L, Dong L, Yuan C, Zukerman M (2007) Fairness comparison of FAST TCP and TCP Reno. Comput Commun 30(6):1375–1382
25. Tan L, Yuan C, Zukerman M (2005) FAST TCP: fairness and queueing issues. IEEE Commun Lett 9(8):762–764
26. Trinh TA, Molnár S (2004) Quality of service in the emerging networking panorama. Lecture notes in computer science, vol 3266, chapter A. Game-theoretic analysis of TCP Vegas. Springer, New York, pp 338–347
27. Trinh TA, Sonkoly B, Molnár S (2004) A HighSpeed TCP study: observations and re-evaluation. In: Proceedings of 10th Eunice summer school and IFIP workshop on advances in fixed and mobile networks, Tampere
28. Wei DX, Jin C, Low SH, Hegde S (2006) FAST TCP: motivation, architecture, algorithms, performance. IEEE/ACM Trans Netw (ToN) 14(6):1246–1259
29. Weigle MC, Sharma P, Freeman J (2006) Performance of competing high-speed TCP flows. In: Proceedings of IFIP networking 2006, Coimbra
30. Welzl M (2005) Network congestion control: managing internet traffic. Wiley, New York