Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Queuing dynamics and single-link stability of delay-based window congestion control

Yueping Zhang [a,*], Yong Xiong [b], Steve Liu [c], Dmitri Loguinov [c]

[a] NEC Laboratories America, Inc., Princeton, NJ 08540, United States
[b] OPNET Technologies, Inc., Bethesda, MD 20814, United States
[c] Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843, United States

ARTICLE INFO

Article history:
Received 21 February 2008
Received in revised form 13 August 2009
Accepted 21 October 2009
Available online 31 October 2009
Responsible Editor: S. Mascolo

Keywords:
Self-clocking
Queuing model
Delay-based congestion control
Asymptotic stability
FAST TCP

ABSTRACT

Accurate modeling of queueing dynamics is important in the design and analysis of Internet congestion control. However, as demonstrated in this paper, existing window-based queueing models [26,30] are often not capable of precisely capturing the transient behavior (i.e., self-clocking and burstiness) of TCP-like protocols and their resulting analysis may be inaccurate in practice. As one example, we show that stability conditions of FAST TCP based on traditional queuing models [17] that do not considering transient dynamics of the queues are inconsistent with ns2 simulations. We explain the origin of this problem and overcome it by developing a novel approach called Self-clocking Queuing Model (SQM) that accurately describes both the steady-state and transient queuing behavior of window-based control systems. Using SQM and explicitly incorporating control interval h_i in the queuing model and derive a sufficient condition for its local stability under homogeneous delay, which strengthens prior results [29,30] obtained using traditional queuing models.

1. Introduction

A significant advance in designing congestion control algorithms for high bandwidth-delay product networks has been made in the last decade. Many recently proposed methods, such as HSTCP [7], STCP [21], FAST TCP [17], XCP [20], BIC TCP [35], and VCP [34], employ *window-based* flow control. Due to the difficulties in modeling various inherent characteristics (such as self-clocking and burstiness) of window-based protocols, better modeling of the system dynamics has gained a renewed interest in the recent literature [13,26,29,30] and now plays an important role in analysis and optimization of window-based control systems.

It is well known that self-clocking is one of the most fundamental properties of window-based protocols [27], in which a new packet is not sent out until an old one is acknowledged. However, most existing work on the dynamics of window-based systems is based on the fluid model proposed in [26], which does not model the interaction between end-user behavior and queuing dynamics and therefore is incapable of representing the *closed-loop* nature of self-clocking. To our knowledge, the only model explicitly taking self-clocking into account is introduced in [30]. However, this model utilizes *static* algebraic equations to describe the stationary state inside the router and cannot capture the *dynamic* queueing behavior. In addition, both models do not consider the transient dynamics during the change of the congestion window, in which case the source sends out packets in bursts in a short period of time. As shown in ns2 simulations, both models exhibit non-negligible modeling errors of TCP during transient phases (see below for more). Although it is commonly assumed that the effect of the transient phase on study of a system's dynamics is negligible compared to that of the entire course of the system's evolution (e.g. [30]), we demonstrate on the contrary that a model without taking

* Corresponding author. Tel.: +1 609 951 2647; fax: +1 609 951 2480.
E-mail addresses: yueping@nec-labs.com (Y. Zhang), yxiong03@gmail.com (Y. Xiong), liu@cse.tamu.edu (S. Liu), dmitri@cse.tamu.edu (D. Loguinov).

1389-1286/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2009.10.011

transient behavior into account may inaccurately describe a system's dynamics and the resulting analysis may deviate from the behavior of the physical system.

Driven by this motivation, we propose a new framework called Self-clocking Queuing Model (SQM), which overcomes the limitations of the previous models as well as preserving their strengths. In particular, by introducing the concept of "pseudo-queue", SQM independently models the behavior of *individual* end-flows and couples it with the queuing dynamics of routers in the user's path. This way, the resulting closed-loop model is able to precisely describe not only the steady-state self-clocking property, but also the phenomenon of bursty packets transmissions of window-based schemes. Similarly to both existing queuing models, SQM assumes a sufficiently large buffer with zero packet loss and does not take into account arriving/departing flows or reverse traffic. As demonstrated in the paper, SQM outperforms previous models and matches ns2 simulations in *both* transient and steady-states. Although applicable to all window-based congestion control systems, we note that SQM can directly benefit dynamics analysis of delay-based algorithms (e.g., TCP Vegas [3] and FAST TCP [17]), since queuing dynamics is explicitly used in end-flow equations of these methods. In the second half of this paper, we demonstrate an application of SQM to stability analysis of FAST TCP [17] in single-link networks with homogeneous feedback delays. Specifically, we derive a sufficient local stability condition, which explicitly captures the effect of control interval h on the system's stability. We also verify tightness of the derived stability condition using ns2 simulations.

The rest of this paper is organized as follows. In Section 2, we provide the background of window-based congestion control and the existing queuing models of these protocols and we discuss the properties and limitations of current queuing models. In Section 3, we present SQM and validate it via ns2 simulations. In Section 4, we apply SQM to local stability analysis of FAST TCP in single-link networks with homogeneous delay. We conclude our work and suggest directions for future research in Section 5.

2. Background

We start by describing the notations used throughout the paper. Assume that a set S of N users are in the network and their congestion windows and sending rates at time t are given by vectors $\mathbf{w}(t) = [w_1(t), \ldots, w_N(t)]$ and $\mathbf{x}(t) = [x_1(t), \ldots, x_N(t)]$, respectively. Further assume that the round-trip time (RTT) and the round-trip propagation delay of flow i are respectively denoted by $R_i(t)$ and D_i, and the forward/backward propagation delays of user i to/from router l by $D^{\rightarrow}_{i,l}$ and $D^{\leftarrow}_{i,l}$, correspondingly. The aggregate arrival rate of users accessing router l is denoted by

$$X_l(t) = \sum_{i \in l} x_i(t - D^{\rightarrow}_{i,l}), \tag{1}$$

where $i \in l$ is the set of flows i passing through link l. Similarly, notation $l \in i$ refers to the set of links l used by flow i.

2.1. Delay-based window congestion control

End-to-end window congestion control algorithms adaptively adjust the congestion window according to certain congestion indicating mechanism, based on which we can classify them into two categories: loss-based and delay-based. Specifically, in loss-based protocols, such as TCP Reno, HSTCP [7], STCP [21], BIC TCP [35], TCP Westwood [31], TCP-Africa [22], and H-TCP [23], senders continually increase their congestion windows until the bottleneck link is saturated and a packet drop is detected. On the other hand, delay-based schemes, such as TCP Tri-S [32,33], TCP Vegas [3], TCP-BFA [1], and FAST TCP [17], utilize round-trip delay as the congestion signal and seek to avoid overflowing router buffers and maintain the queue length around a certain optimal operating point.

The most representative delay-based scheme is FAST TCP, which utilizes queueing delay as the main congestion indication and adjusts the congestion window proportionally to the queueing backlog error [18]. The queuing backlog error is defined as the difference between sender's aggregate queuing backlog along its path and some target value. FAST TCP periodically updates the congestion window $\mathtt{w}_{\mathtt{i}}$ according to

$$\mathtt{w}_{\mathtt{i}} \leftarrow (1-\gamma_i)\mathtt{w}_{\mathtt{i}} + \gamma_i\left(\frac{\mathtt{baseRTT}_{\mathtt{i}}}{\mathtt{RTT}_{\mathtt{i}}}\mathtt{w}_{\mathtt{i}} + \alpha_i\right), \tag{2}$$

where $\mathtt{RTT}_{\mathtt{i}}$ is the estimated round-trip delay of user i and is denoted by $R_i(t)$ in the rest of the paper, $\mathtt{baseRTT}_{\mathtt{i}}$ is the observed minimum $\mathtt{RTT}_{\mathtt{i}}$, γ_i is feedback control gain, and α_i is the target queue length. Approximating the round-trip propagation delay D_i with $\mathtt{baseRTT}_{\mathtt{i}}$ and sending rate $x_i(t)$ with the quotient between congestion window $w_i(t)$ and RTT $R_i(t)$ as shown in (5), the window dynamics can further be described by the following difference equation [18]:

$$w_i(t+1) = w_i(t) - \gamma_i(p_i(t)x_i(t) - \alpha_i), \tag{3}$$

where the aggregate queueing delay $p_i(t)$ of user i is given by (8). The corresponding continuous-time version of (3) is thus given below [18]:

$$\dot{w}_i(t) = -\gamma(p_i(t)x_i(t) - \alpha_i). \tag{4}$$

Note that in (4) the time is measured in the units of update intervals. Here, term $p_i(t)x_i(t)$ can be interpreted as an approximation of the number of flow i's packets buffered in the path. Without routers' explicit feedback, estimating the actual queue length in the Internet is very difficult. Thus, similar to FAST, several protocols (e.g., TCP Vegas [3], TCP Illinois [24], and PERT [2]) rely on queuing delay measurements to infer queue backlogs.

It is proved that in the absence of delay and $\gamma_i \in (0,1]$, FAST TCP is locally stable in multi-link networks [17,30] and globally stable in a single-link network [30]. When the feedback delay is present and under the same condition, Wang et al. [30] prove that FAST TCP is locally stable as long as all flows have the same round-trip delay.

2.2. Modeling of queuing dynamics

To fit the analytical framework of the control theory to the congestion control analysis, one can treat the queuing dynamics as the *controlled plant*, and the window adjustment laws as the *controllers* of a closed-loop control system. However, precisely modeling *window-based* congestion control systems appears to be difficult due to their intrinsic properties, such as self-clocking and bursty packets transmission. In this section, we review the existing work on modeling queuing dynamics for window-based schemes.

Most theoretical models of window-based congestion control schemes in the literature [4,13,17,20,26,29,30] are based on the common approximation of the sending rate $x_i(t)$ of source i as:

$$x_i(t) = \frac{w_i(t)}{R_i(t)} \tag{5}$$

and the queuing dynamics of link l as:

$$\dot{q}_l(t) = \begin{cases} X_l(t) - C_l & \text{if } q_l(t) > 0 \text{ or } q_l(t) = 0,\ X_l(t) > C_l, \\ 0 & \text{otherwise}, \end{cases} \tag{6}$$

where the aggregate input rate $X_l(t)$ is given by (1), $q_l(t)$ is the queue length at link l, and C_l is the link capacity. For ease of presentation, we refer to (5) and (6) as *Model I*.

Another queuing model is proposed in [30], where the queuing dynamics are explicitly considered when modeling the end-user behavior. Specifically, first re-write the round-trip delay $R_i(t)$ of source i as the summation of its round-trip propagation delay D_i and the end-to-end queuing delay $p_i(t)$, i.e.,

$$R_i(t) = D_i + p_i(t), \tag{7}$$

where $p_i(t)$ is the aggregate queuing delay of the set of links in the path of source i, that is

$$p_i(t) = \sum_{l \in i} \frac{q_l\left(t - D^{\leftarrow}_{i,l}\right)}{C_l}. \tag{8}$$

We note that, in the presence of reverse congestion, backward delay is composed of the constant propagation delay and random queuing delay. In this case, the last equation still holds except that $D^{\leftarrow}_{i,l}$ is replaced by $D^{\leftarrow}_{i,l}(t)$. In the paper, we assume there is no traffic in the reverse path, implying that ACKs are not subject to backward queuing delays and are retarded by time-invariant backward delay $D^{\leftarrow}_{i,l}$ for the rest of the paper. Then, the queuing dynamics of any link l in the system can be described as follows [30]:

$$\sum_{i \in S_l} \frac{w_i\left(t - D^{\rightarrow}_{i,l}\right)}{D_i + p_i(t - D^{\rightarrow}_{i,l})} \begin{cases} = C_l, & \text{if } q_l(t) > 0, \\ \leqslant C_l, & \text{if } q_l(t) = 0. \end{cases} \tag{9}$$

We call Eq. (5) and (7)–(9) as *Model II*.

As discussed below (Section 2.3), both Models I and II have certain limitations in capturing the ACK-clocking behavior of TCP. To mitigate this problem, another queuing model is introduced in [14–16,28]:

$$\dot{p}(t) = \frac{1}{c}\left(\sum\nolimits_{i=1}^{N}\left(\frac{w_i(t - D^{\rightarrow}_i)}{D_i + p(t)} + \dot{w}_i(t - D^{\rightarrow}_i)\right) - c\right). \tag{10}$$

In the last equation, term $\dot{w}_i(t - D^{\rightarrow}_i)$ describes instantaneous queuing dynamics due to window changes, thereby improving Models I and II in capturing TCP's bursty packet transmission. This model is referred to as *Joint Model* in that it can be seen as a superposition of Models I and II [15]. As demonstrated in [14], in contrast to Models I and II, Joint Model is able to closely describe queuing dynamics under abrupt changes in congestion window, which is referred to as *sub-RTT effect* and is equivalent to the *step response* test conducted later in Section 3.4.

In the next section, we seek to gain more insight into these queuing models and identify their limitations.

2.3. Discussion of existing queuing models

Model I describes the system's *average* behavior, where source i sends out $w_i(t)$ packets during one round-trip delay $R_i(t)$. In other words, given a fixed congestion window, this model approximates the "pacing" behavior of TCP in the steady-state. However, during the *transient* phase of a congestion window change, the source sends out packets in bursts, and the packets transmission will be clocked at the packet rate of the ACK one RTT after this change. Clearly, this phenomenon is not captured by the steady-state equation (5) of Model I. This is because when the router receives a burst of packets, its queuing dynamics are described in the differential equation (6); however, this effect is not reflected in sources' packets transmission behavior in Model I, which does not provides an explicit model connecting the end-user behavior (5) and queuing dynamics (6). Note that this open control loop of Model I in fact can be closed by the implicit relationship $\dot{R}_i = \dot{p}_i$, which however is not included in the model or used in the associated stability analysis. This problem is addressed by Model II.

On the other hand, Model II, in (7), extracts from the round-trip delay $R_i(t)$ the propagation delay D_i and queuing delay $p_i(t)$. Note that $p_i(t)$ not only describes queuing dynamics in (9) but is further utilized in the end-user equation (5). This constructs a closed-loop model, which leads to a more precise description of the system's dynamics and, more importantly, a better framework for modeling the self-clocking of window-based congestion control systems. However, a drawback of (9) is that it uses static algebraic expressions instead of the dynamic differential equations as in (6) to describe the queuing behavior. This results in ineffectiveness in modeling transient system dynamics. In addition, (8) calculates queuing delay based on $q_l(t - D^{\leftarrow}_{i,l})$, which is the aggregate queue size and does not capture the queuing dynamics of individual flows. Namely, it still characterizes the *average* dynamics of the system and does not provide an accurate model of individual end-users.

As we demonstrate via `ns2` simulations later in the paper, both models exhibit non-negligible modeling errors in transient phases. In this paper, we propose a new queuing model called Self-clocking Queuing Model (SQM) that captures both steady-state and transient dynamics of the router's queuing behavior. As an application, we demonstrate that using SQM, one can improve the single-link stability analysis of FAST TCP under homogeneous delay. We remark that the Joint Model is in spirit similar to SQM, but their underlying theoretical foundations are very different. In addition, as it becomes clear later, SQM leads to less conservative stability conditions of FAST TCP than the Joint Model [15,28] and allows for accurate analysis of the impact of control interval on the system's stability.

Finally, we note that another innovative methodology for analyzing performance of window-based transport protocols is *queuing networks*, in which, based on a finite-state machine description of TCP, a closed queuing network model is constructed to analyze the behavior of several TCP flavors (e.g., Tahoe, Reno, and SACK). Using this powerful technique, one can estimate dynamics of both short- [9] and long-lived [8] TCP sessions, model interactions of a large number of flows [5,10], and predict performance of certain modifications to the TCP protocol [6]. Compared to this queuing-networks based method, the above control-theoretic approaches offer an alternative way to modeling the interaction between TCP and router buffer and are particularly useful for determining key control parameters to achieve desired system stability.

3. Self-clocking Queuing Model (SQM)

3.1. Pseudo-queue

Let $c_l(t)$ be the rate of packets departing from link l at time t. To model the behavior of individual flows, we approximate the departure rate $c_{li}(t)$ of each flow i (i.e., the average rate at which packets of flow i leaves the router). It is easy to see that when link l is under-utilized, flow i's departure rate is same as its arrival rate. Next consider the case when link l is saturated. Note that the combined departure rate $c_l(t)$ now equals the link capacity C_l. Let $q_{li}(t)$ be the number of flow i's packets buffered in the router l at time t. We call $q_{li}(t)$ the length of flow i's *pseudo-queue* in router l. Define $\rho_i = q_{li}(t)/q_l(t)$ as the proportion of flow i's packets in the buffer. Assuming a fluid model of each flow, among the $C_l\Delta t$ packets that depart from router l during a small time interval $[t, t+\Delta t]$, the ratio of packets belonging to flow i is ρ_i. Thus, flow i's departure rate can be represented by

$$c_{li}(t) = \rho_i C_l. \tag{11}$$

Replacing ρ_i with $q_{li}(t)/q_l(t)$ in the last equation, we have

$$q_{li}(t) = c_{li}(t)\frac{q_l(t)}{C_l}. \tag{12}$$

Clearly, the pseudo-queue lengths $q_{li}(t)$ of all flows accessing link l sum up to the actual queue length $q_l(t)$ of this link:

$$\sum_{i\in S_l} q_{li}(t) = q_l(t). \tag{13}$$

In addition, the end-to-end queuing delay of flow i can be expressed by the summation of the queuing delay of each router in its path, i.e.,

$$p_i(t) = \sum_{l\in i}\frac{q_{li}\left(t-D^{\leftarrow}_{i,l}\right)}{c_{li}\left(t-D^{\leftarrow}_{i,l}\right)}, \tag{14}$$

where $l \in i$ denotes the set of routers in flow i's path. It is easy to notice that (14) is essentially equivalent to (8); however, the former reflects the queuing behavior of *individual* session flows and, as demonstrated later in this section, results in an explicit closed-form expression of the dynamics of each flow i. It is also evident that the total output rate $c_l(t)$ of link l is constrained by the link capacity, that is,

$$c_l(t) = \sum_{i\in S_l} c_{li}(t) \leqslant C_l. \tag{15}$$

As we show later, this model precisely captures the self-clocking property of window-based schemes and the bursty nature of TCP-like traffic. Thus, we call this model Self-clocking Queuing Model (SQM). However, we note that, similar to Models I and II, SQM assumes a sufficiently large buffer and does not consider packet loss, which is an important factor in modeling loss-based window congestion control (e.g., TCP Reno). Thus, in this paper, we limit our discussion to delay-based protocols, such as TCP Vegas and FAST TCP, and leave extensions of SQM to generic window congestion control for future work. In addition, SQM also does not take into account arrival/departure of flows, short-lived congestion-unresponsive sessions, or reverse traffic. Thus, extending SQM to such dynamic environments forms another line of our future work.

We further remark that constructing a unified queuing framework for multi-link topologies is very difficult, since a separate differential equation is necessary to capture queuing dynamics of each link in the path and an additional model is required to describe interaction between queuing behavior of individual links. On the other hand, in the route of any particular user, there is only one bottleneck link, which is responsible for most of the queuing dynamics and delay of the entire path. We note that in practice the bottleneck link of a particular flow can shift over time due to transient dynamics (e.g., route changes and arrival/departure of cross traffic) of the network [36]. It is a common assumption made by all existing queuing models [26,30] and the resulting stability analysis [13,20,29,30,34] that the bottleneck assignment is time-invariant. In addition, most of these works consider networks where all flows share a single bottleneck link. In the rest of this section, we follow this common practice and study SQM in single-link scenarios and compare its performance in modeling dynamics of TCP with traditional queuing models via `ns2` simulations.

3.2. Single-link SQM

For the single-link case, we omit the router subscript l for ease of presentation. Assuming $y_i(t)$ is the ACK packet rate perceived by source i and given a small time period $[t, t+\Delta t]$, the user receives $y_i(t)\Delta t$ ACKs during this interval and therefore sends out $y_i(t)\Delta t$ packets due to the self-clocking property. Here, we assume that one ACK acknowledges exactly one data packet. Note that in practice ACK rate $y_i(t)$ can be bursty in the presence of reverse traffic [12]. As stated earlier, we assume in the paper that there is no traffic in the reverse path, in which case we believe that constant $y_i(t)$ is a valid approximation for small time interval Δt. In addition, as in TCP Westwood+ [25], ACK rate is used to infer the available bandwidth, based on which the sending host dynamically sets its slow-start threshold and congestion window size. We do not model this behavior in the paper.

Assume during interval $[t, t+\Delta t]$ source i changes its congestion window by $\Delta w_i(t)$ (which can be both positive and negative), the total number of packets sent out during interval $[t, t+\Delta t]$ is approximately $y_i(t)\Delta t + \Delta w_i(t)$. Hence, we can write the sending rate $x_i(t)$ of user i as the following fluid model:

$$x_i(t) = y_i(t) + \dot{w}_i(t). \tag{16}$$

In a single-link network, the rate at which user i's packets depart from the router equals the ACK rate perceived by the user after one backward delay $D_i^{\leftarrow}$ plus queuing delay q_i, i.e., $y_i(t) = c_i(t - D_i^{\leftarrow} - q_i)$. This translates (16) into:

$$x_i(t) = c_i\big(t - D_i^{\leftarrow} - q_i\big) + \dot{w}_i(t). \tag{17}$$

Analogous to (6), we can describe the evolution of the pseudo-queue by:

$$\dot{q}_i(t) = x_i\big(t - D_i^{\rightarrow}\big) - c_i(t), \tag{18}$$

which, combined with (17), is transformed into:

$$\dot{q}_i(t) = \dot{w}_i\big(t - D_i^{\rightarrow}\big) + c_i(t - R_i) - c_i(t) \tag{19}$$

and its discrete-time counterpart:

$$\begin{aligned} q_i(n + h_i) = q_i(n) &+ w_i\big(n - D_i^{\rightarrow} + h_i\big) - w_i\big(n - D_i^{\rightarrow}\big) \\ &+ c_i(t - R_i) - c_i(t), \end{aligned} \tag{20}$$

where R_i is the round-trip delay of user i. Taking the summation of (19) over all flows and using (13), we obtain the following differential equation for the overall queuing dynamics:

$$\dot{q}(t) = \sum_{i\in S}\dot{w}_i\big(t - D_i^{\rightarrow}\big) + \sum_{i\in S} c_i(t - D_i) - c(t), \tag{21}$$

where S denotes the set of flows in the system, C is the bottleneck link capacity, and $c(t)$ is the combined departure rate given in (15).

We next derive the equation for $c_i(t)$, whose value is determined by the following two cases.

- When $q(t) > 0$, we have $c_i(t) = Cq_i(t)/q(t)$ directly from (12).
- When $q(t) = 0$, we have that the combined input rate $X(t) = \sum_{i\in S} x_i\big(t - D_i^{\rightarrow}\big)$ does not exceed the link capacity C. Then, flow i's output rate $c_i(t)$ equals its input rate $x_i\big(t - D_i^{\rightarrow}\big)$ at the bottleneck link. Combining this observation, (17), and the fact that $c_i(t) \geqslant 0$ leads to $c_i(t) = c_i(t - D_i) + \dot{w}_i\big(t - D_i^{\rightarrow}\big)$.

We summarize the above results as follows:

$$c_i(t) = \begin{cases} \frac{q_i(t)}{q(t)}C & \text{if } q(t) > 0, \\ \big(\dot{w}_i\big(t - D_i^{\rightarrow}\big) + c_i(t - D_i)\big)^{+} & \text{if } q(t) = 0, \end{cases} \tag{22}$$

where $(x)^{+} = \max(0, x)$.

Eqs. (19) and (22) comprise the single-link SQM under directional feedback delays. Note that this system explicitly describes the behavior of *individual* flows and the coupling between different flows comes from the second and third terms in (19). It is clear that this system is a nonlinear closed-loop feedback control system. The second term of (19) is the retarded version of the third term. Applying the functional differential equation theory [11] to (19) and (22), we can arrive at some elegant results. As it will become clear later, these results significantly facilitate asymptotic stability analysis of window-based congestion control protocols, such as FAST TCP. Before proceeding in that direction, we next elaborate on some of the properties of SQM and compare it with Models I and II using `ns2` simulations.

3.3. Properties of SQM

First, the single-link SQM consisting of (19) and (22) has *constant* (i.e., time-invariant) delay because neither D_i nor $D_i^{\rightarrow}$ includes the queuing delay when there is only one bottleneck link. Second, SQM captures the bursty nature and the self-clocking property of window-based schemes. Specifically, the first term $\dot{w}_i\big(t - D_i^{\rightarrow}\big)$ in (19) is the control input. Thus, an abrupt change of congestion windows will cause an abrupt change of the queue length after a forward delay $D_i^{\rightarrow}$. This is also reflected in the aggregate queueing dynamics (21). The second term $c_i(t - D_i)$ in (19) captures the self-clocking property of individual flows because $c_i(t - D_i) = y_i\big(t - D_i^{\rightarrow}\big)$ and $y_i(t)$ is the ACK packet rate.

Third, the following well-known properties of window-based schemes are also accommodated in this model.

Lemma 1. *Let $F_i(t)$ denote the number of source i's packets that are on the way to the destination at time t. For a single-link SQM given by (19) and (22), if at time 0 the congestion window size $w_i(0)$ of source i equals the number of outstanding packets $F_i(0)$, then this condition continues to hold for all subsequent time, i.e., $F_i(t) = w_i(t)$ for $t \geqslant 0$.*

Proof. First note that as illustrated in Fig. 1, the set of packets in flight consist of the packets buffered in the queue and those in the forward and backward paths. Then, we have

$$F_i(t) = q_i(t) + \int_{t-D_i^{\rightarrow}}^{t} x_i(\tau)d\tau + \int_{t-D_i^{\leftarrow}}^{t} c_i(\tau)d\tau. \tag{23}$$

Substituting (17) into (23), we get

$$F_i(t) = q_i(t) + \int_{t-D_i}^{t} c_i(\tau)d\tau + \int_{t-D_i^{\rightarrow}}^{t} \dot{w}_i(\tau)d\tau. \tag{24}$$

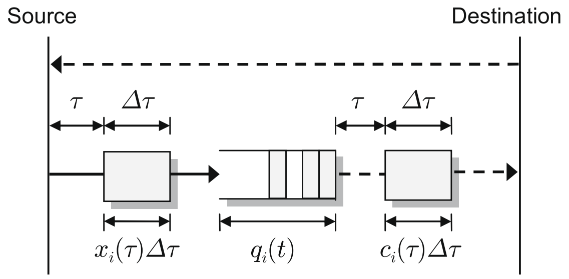


Fig. 1. The number of packets of source i that are being transmitted.

Taking derivatives of both sides of the last equation and using (19), we have $\dot{F}_i(t) = \dot{w}_i(t)$. Since $F_i(0) = w_i(0)$ as assumed in the lemma, we directly arrive at $F_i(t) = w_i(t)$ and complete the proof. □

Note that the above lemma considers only the *steady-state* behavior of window-based congestion control. In transient phases where congestion window suddenly reduces, it takes at least one RTT before the number of outstanding packets becomes equal to the congestion window again. The next conclusion is straightforward.

Lemma 2 (Conservation property). *In a single-link SQM given by (19) and (22), the number of outstanding packets is constant when all congestion windows are constant.*

Proof. Since $\dot{F}_i(t) = \dot{w}_i(t)$ by Lemma 1, the result directly follows. □

Using simple manipulations, we can further get the following result based on Lemma 1.

Lemma 3. *In a single-link SQM given by (19) and (22), the stationary sending rate x_i^* of source i is $x_i^* = w_i^*/R_i^*$.*

Proof. Substituting $F_i(t) = w_i(t)$ into (24), we get

$$w_i(t - D_i^{\rightarrow}) = q_i(t) + \int_{t-D_i}^{t} c_i(\tau)d\tau. \tag{25}$$

Thus, in the steady-state, we get $w_i^* = q_i^* + c_i^* D_i$. Using (12), we have

$$c_i^*\left(D_i + \frac{q^*}{C}\right) = c_i^*(D_i + p_i^*) = w_i^*. \tag{26}$$

Noticing that $R_i^* = D_i + p_i^*$ and $x_i^* = c_i^*$ in the steady-state, we get $x_i^* = w_i^*/R_i^*$. □

Note that, in the transient state, the sending rate $x_i(n)$ of source i is not necessarily equal to $w_i(n)/R_i(n)$ due to the bursty packets transmission.

In summary, we conducted a series of simple analytical validations of SQM and confirms its capability of capturing several basic properties of window-based congestion control schemes. To gain a better understanding of SQM, we next present its ns2 simulations and compare its performance to that of Models I and II based on their accuracy in describing the self-clocking and burstiness properties of TCP traffic.

3.4. Model validation

The network configuration for the simulation is illustrated in Fig. 2, where three flows share a single bottleneck link of capacity 10 Mb/s. The round-trip propagation delays of these three sources are 420 ms, 460 ms and 432 ms, respectively. We set the buffer size sufficiently large to avoid the influence of queue overflow. This allows us to focus on the performance of SQM and traditional models (i.e., Models I and II) in modeling queuing dynamics of a buffer fed by TCP's self-clocked and bursty traffic.

To compare these three models, we evaluate their ability to characterize the system's *step* response and *ramp* response to changes in the congestion window.

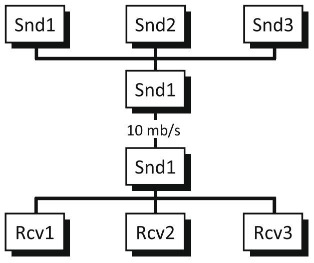


Fig. 2. Simulation topology.

Step response: This experiment is to demonstrate the system's response to a *sudden* increase in the congestion window, which is also called *sub-RTT effect* as in [14]. Specifically, we use the following input step function to emulate the case where the congestion windows of all three sources jump from 0 to 400 packets at time 0:

$$w_i(t) = 400H(t)\ \text{pkt}, \tag{27}$$

where $H(t)$ is a unit step function, which equals to 0 for $t < 0$ and 1 for $t \geqslant 0$. Fig. 3 shows that Models I and II match ns2 simulation results in the steady-state; however, both of them have large errors in the transient stage during the first second. In contrast, SQM matches ns2 simulations very well in both steady and transient states.

Ramp response: In this experiment, we examine all these three models' capability of capturing the system's response to *continuous* increase in congestion window. Specifically, the congestion windows of all three flows keep increasing at a speed of 300 pkt/s. The control input can be expressed as:

$$\dot{w}_i(t) = 300\ \text{pkt/s}. \tag{28}$$

Fig. 4a shows the time histories of the queue length under the control input (28). It shows that Model I matches the ns2 simulation very well except within a brief period when the queue length begins to build up; Model II exhib-

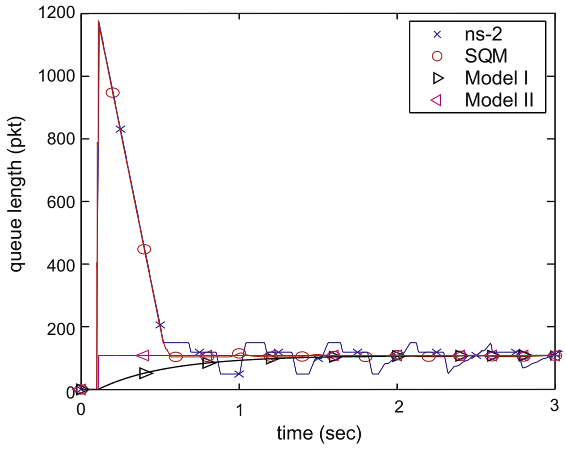
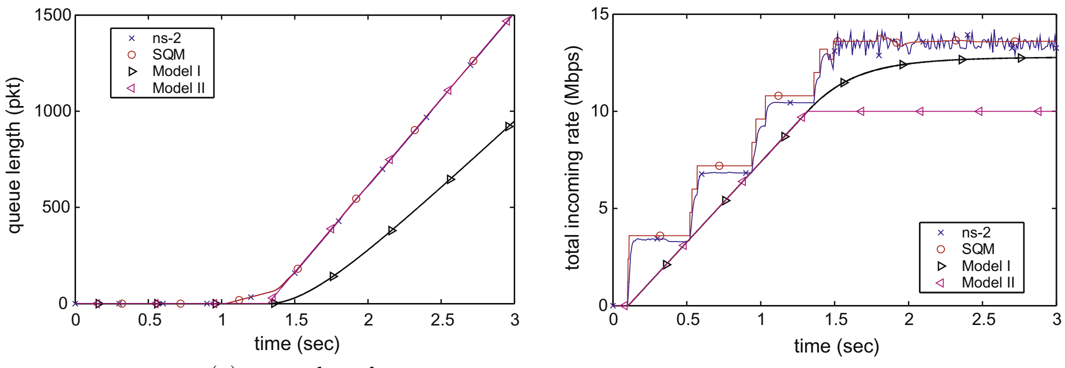


Fig. 3. Impulse response.



(a) queue length (b) combined incoming rate

Fig. 4. Step response.

its a significant discrepancy from the ns2 simulation; and SQM matches the ns2 simulation very well in all stages.

In addition, we plot the total incoming rate at the bottleneck link in Fig. 4b. As shown in the figure, SQM matches the ns2 simulation much better than Models I and II. Notice that this result shows that the total input rate performs like a staircase before the queue builds up. During the first RTT, the sending rate of each flow equals the window variation rate $(300 \times 8 \times 500 = 1.2\ \text{Mb/s})$ since there are no ACK packets received. Thus, the total incoming rate is $1.2 \times 3 = 3.6$ Mb/s within the fist RTT. During the second RTT, due to the self-clocking property, the sending rate equals the window variation rate (1.2 Mb/s) plus the ACK packet rate (1.2 Mb/s). Thus, during the second RTT, the sending rate equals 2.4 Mb/s and the total incoming rate is 7.2 Mb/s. Similarly, we can get the sending rate during the third RTT. After 1.5 s, the queue starts to build up, which causes the ACK packet rate of each flow to roughly equal 10/3 = 3.33 Mb/s. Therefore, the total incoming rate is $10 + 1.2 \times 3 = 13.6$ Mb/s after 1.5 s. As illustrated in the figure, only SQM captures such step-like behavior of the total incoming rate.

Collectively, in both experiments, SQM captures the transient queuing dynamics much more precisely than the other two models. Even though it is customarily assumed that modeling of the system dynamics in the transient phase is less important compared to that in the equilibrium state, as we show in the next section, this assumption does not hold in practice and can lead to incorrect conclusions of a system's steady-state behavior.

In the rest of the paper, we show an application of SQM to stability analysis of FAST TCP. We note that stability of FAST has been intensively studied. In the following section, we do not seek to gain a significant advance in this frontier, but demonstrate how to apply SQM to a practical system to obtain improved stability results (e.g., explicit modeling of control interval h_i) with a simpler process.

4. FAST TCP under homogeneous delay

In the rest of the paper, we apply SQM to stability analysis of FAST TCP and demonstrate how one can improve previous results using a simple reasoning process.

4.1. Modeling FAST TCP

We use n and t respectively to denote the time variable in the discrete- and continuous-time model, and h_i to represent the control interval of end-user i. We note that all previous work [15,16,28] on stability of FAST all implicitly assume h_i equals one RTT of user i, while in this paper we *explicitly* incorporate h_i in our model and study its impact on the system's stability. Using SQM, FAST TCP's end-user Eq. (3) can be re-written as[1]

$$w_i(n+h_i) = w_i(n) - \gamma_i(p_i(n)x_i(n) - \alpha_i) \tag{29}$$

and the aggregate queuing delay $p_i(n)$ given by (14) becomes

$$p_i(n) = \sum_{l \in i} \frac{q_{li}\left(n - D^{\leftarrow}_{i,l}\right)}{c_{li}\left(n - D^{\leftarrow}_{i,l}\right)}. \tag{30}$$

In the steady-state, α_i equals the number of flow i's packets buffered in the routers along the path of i [18]. From the viewpoint of feedback control, the control objective of (3) is to drive the term $p_i(n)x_i(n)$ to the target value α_i. Notice that $p_i(n)x_i(n)$ is an approximation of the number of flow i's packets buffered in the routers along its path, i.e., $p_i(n)x_i(n) \approx \sum_{l \in i} q_{li}\left(n - D^{\leftarrow}_{i,l} - p_l\right)$, where p_l is the queuing delay at link l and is assumed to be an integer.

Further, we note that throughout the paper we consider FAST TCP only in its operational region where the stationary combined queuing requirement $\sum_i \alpha_i$ does not exceed the network's buffering provision. Thus, the window adjustment dynamics of FAST TCP can be re-written as

$$w_i(n+h_i) = w_i(n) - \gamma_i\left(\sum_{l \in i} q_{li}\left(n - D^{\leftarrow}_{i,l} - p_l\right) - \alpha_i\right) \tag{31}$$

or equivalently the continuous-time version

$$\dot{w}_i(t) = -\eta_i\left(\sum_{l \in i} q_{li}\left(t - D^{\leftarrow}_{i,l} - p_l\right) - \alpha_i\right), \tag{32}$$

where $\eta_i = \gamma_i/h_i$.

[1] Receive-window-limited flows are not modeled in the paper.

For the single-link case, (31) and (32) become

$$w_i(n+h_i) = w_i(n) - \gamma_i(q_i(n - D_i^{\leftarrow} - p_l) - \alpha_i) \tag{33}$$

and

$$\dot{w}_i(t) = -\eta_i(q_i(t - D_i^{\leftarrow} - p_l) - \alpha_i), \tag{34}$$

respectively. Using this model, we next prove stability of FAST TCP under homogeneous delay.

4.2. Stability under homogeneous delay

By homogeneous delay, we mean that all end-users experience the same RTT (i.e., $R_i = R_j = R$ for all $i \neq j$) and adjust their congestion windows at the same time scale (i.e., $h_i = h_j = h$). Define $\phi = R/h$, which is assumed to be an integer. Then, the main result of this section is given in the following theorem.

Theorem 1. *In a single-link network under homogeneous delay, the discrete-time model of FAST TCP described by (20) and (33) with homogeneous γ is locally asymptotically stable if*

$$0 < \gamma < 2\sin\left(\frac{\pi}{2(2\phi+1)}\right) \tag{35}$$

and unstable for certain ϕ if

$$\gamma > 2\sin\left(\frac{\pi}{2(2\phi+1)}\right). \tag{36}$$

Proof. Substituting (33) into (20) with R_i and h_i respectively replaced by their homogeneous values R and h, we can describe the discrete-time queuing dynamics for user i using the following difference equation:

$$q_i(n+h) = q_i(n) - \gamma(q_i(n-R) - \alpha_i) + c_i(n-R) - c_i(n), \tag{37}$$

where $\alpha = \sum_{i \in S} \alpha_i$ is the aggregate target queue length. Representing the last equation in time units of control interval h and recalling that $R/h = \phi$, we have:

$$q_i(m+1) = q_i(m) - \gamma(q_i(m-\phi) - \alpha_i) + c_i(m-\phi) - c_i(m), \tag{38}$$

where each unit of time variable m equals h units of n. Taking summation of the last equation over all flows $i \in S$ and recalling (13) and (15), we have:

$$q(m+1) = q(m) - \gamma(q(m-\phi) - \alpha) + \sum_{i \in S} c_i(m-\phi) - c(m). \tag{39}$$

Note that the stationary queue length q^* of FAST is always positive. Then, to analyze local stability of FAST, we only need to consider $q(m)$ in the neighborhood of q^*, in which case we have $q(m) > 0$ and therefore $\sum_{i \in S} c_i(m-\phi) = c(m-\phi) = c(m) = C$,[2] which transforms (39) to:

$$q(m+1) = q(m) - \gamma(q(m-\phi) - \alpha). \tag{40}$$

Note that (40) is the same as the system studied by Johari et al. [19]. Thus, simply invoking [19, Theorem 1], we reach the conclusion that linear difference equation (40) is locally asymptotically stable if (35) is satisfied and unstable if (36) holds. □

Even though stability of the system when $\gamma = 2\sin(\pi/(2(2\phi+1)))$ is unknown according to Theorem 1, it is evident that (35) represents a *tight* local stability condition of FAST TCP under homogeneous delay. In addition, simulations show that in many practical situations the queue never empties once its size becomes positive, which suggests that system (20) and (33) is in fact *globally* stable in its entire operational region if (35) is satisfied. Moreover, Theorem 1 explicitly incorporates end-flows' control intervals h in the stability condition, which improves previous results [15,16,28,30,29] that hold only if the sender updates its congestion window once per RTT or every two RTTs.

4.3. Result validation

We next validate Theorem 1 using the following `ns2` simulation.

Experiment 1. In a single-link network with a single FAST TCP source, the round-trip propagation delay D is 10 ms, link capacity C is 10 Mb/s, buffer size is 200 packets, packet size is 500 bytes, target queue length α is 50 packets. The source updates its congestion window upon each ACK packet instead of per RTT or every two RTTs as suggested in [17]. Thus, the update interval h of the end-user equals to the inter-packet intervals of the ACKs. Assuming that the packet size is s and noticing that after the queue is built up, the ACK rate $y(n)$ is just the link capacity C, we can approximate the update interval h as follows:

$$h = \frac{s}{C} = \frac{500 \times 8}{10^7} = 0.4\ \text{ms}. \tag{41}$$

In addition, since the target queue length is 50 packets, it is easy to obtain that the steady-state queuing delay p is 20 ms. Then, metric $\phi = R/h = (D+p)/h = 30/0.4 = 75$. According to Theorem 1, the system is asymptotically stable if and only if $\gamma \in (0, 0.0208)$. This prediction is well matched by `ns2` simulations shown in Fig. 5. From the figure, we can see that the system is asymptotically stable when $\gamma = 0.01$, marginally stable when $\gamma = 0.0208$, and unstable when $\gamma = 0.03$, in which case the congestion window exhibits persistent oscillations and periodically drops to zero.

We next consider another example to verify Theorem 1.

Experiment 2. In this experiment, we utilize the "dumb bell" topology shown in Fig. 2 and set the feedback delay to be zero. All other parameters are the same as those in Experiment 1. The congestion windows are updated every ACK. According the Theorem 1, the system should be stable when $0 < \gamma < 2$ and unstable when $\gamma > 2$. As illustrated in Fig. 6, the system is asymptotically stable when $\gamma = \{0.1, 1.0, 1.8\}$ and unstable when $\gamma = 2.1$.

[2] Cross-traffic and congestion-unresponsive flows (e.g., mice) may also lead to queue backlog. We do not model effects of these types of traffic in the paper.

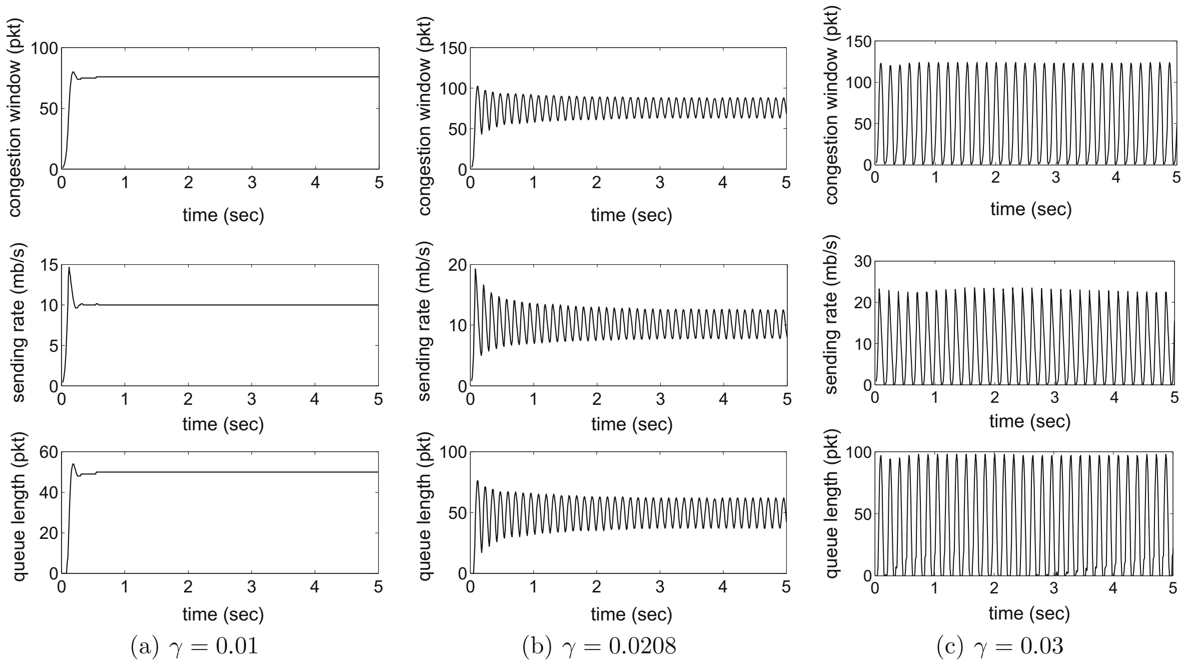


Fig. 5. ns2 simulations for the single-link single-flow case in Experiment 1.

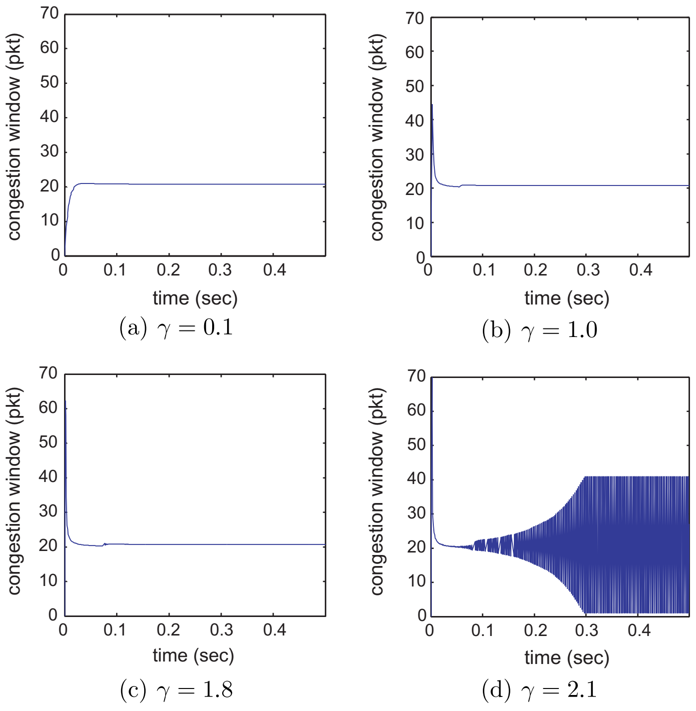


Fig. 6. Congestion window (packet) vs. time (second) in Experiment 2.

5. Conclusion

In this paper, we demonstrated that precise modeling of transient dynamics was an important but most overlooked issue when analyzing a closed-loop congestion control system. Motivated by this finding, we developed a new queueing model called SQM, which precisely characterized the self-clocking and bursty packets transmission of window-based congestion control in both the transient and steady-states. The subtle structures of the nonlinear terms of SQM allowed us to take advantage of the functional differential equation theory and obtain precise stability conditions without taking any approximation terms. We limited stability analysis to FAST TCP, while with proper incorporation of the various control laws, SQM can also be used to analyze other window-based congestion control algorithms such as XCP, Vegas, and Reno. In addition, the current application of SQM is restricted to the single-link cases and we leave its extension to more generic scenarios for the future work. Other improvements of SQM involve modeling the effect of mice traffic, congestion in the reverse path, and time-varying delays.

References

[1] A.A. Awadallah, C. Rai, TCP-BFA: buffer fill avoidance, in: Proc. IFIP HPN, September 1998, pp. 575–594.

[2] S. Bhandarkar, A.L.N. Reddy, Y. Zhang, D. Loguinov, Emulating AQM from end hosts, in: Proc. ACM SIGCOMM, August 2007, pp. 349–360.

[3] L. Brakmo, S. O'Malley, L. Peterson, TCP Vegas: new techniques for congestion detection and avoidance, in: Proc. ACM SIGCOMM, August 1994, pp. 24–35.

[4] H. Choe, S.H. Low, Stabilized vegas, in: Proc. IEEE INFOCOM, April 2003, pp. 2290–2300.

[5] R.L. Cigno, M. Gerla, Modeling window based congestion control protocols with many flows, Comput. Netw. 36 (1) (1999) 289–306.

[6] R.L. Cigno, G. Procissi, M. Gerla, Sender-Side TCP Modifications: Performance Analysis and Design Guidelines, Cluster Comput. 8 (1) (2005) 35–45.

[7] S. Floyd, High-speed TCP for large congestion windows, IETF RFC 3649, December 2003.

[8] M. Garetto, R.L. Cigno, M. Meo, M.A. Marson, Closed queuing network models of interacting long-lived TCP flows, IEEE/ACM Trans. Netw. 12 (2) (2004) 300–311.

[9] M. Garetto, R.L. Cigno, M. Meo, M.A. Marson, Modeling short-lived TCP connections with open multiclass queuing networks, Comput. Netw. 44 (2) (2004) 153–176.

[10] M. Garetto, R.L. Cigno, M. Meo, M.A. Marson, A detailed and accurate closed queueing network model of many interacting TCP flows, in: Proc. IEEE INFOCOM, December 2001, pp. 1706–1715.

[11] P. Glendinning, Stability, Instability and Chaos: An Introduction to the Theory of Nonlinear Differential Equations, Cambridge University Press, 1994.

[12] L.A. Grieco, S. Mascolo, Performance evaluation and comparison of Westwood+ New Reno and Vegas TCP congestion control, ACM SIGCOMM Comput. Commun. Rev. 34 (2) (2004) 25–38.

[13] C.V. Hollot, V. Misra, D. Towsley, W. Gong, Analysis and design of controllers for AQM routers supporting TCP flows, IEEE Trans. Automat. Contr. 47 (6) (2002) 945–959.

[14] K. Jacobsson, L.L.H. Andrew, A. Tang, K.H. Johansson, H. Hjalmarsson, S. Low, ACK-clocking dynamics: modelling the interaction between windows and the network, in: Proc. IEEE INFOCOM, April 2008, pp. 181–185.

[15] K. Jacobsson, L.L.H. Andrew, A.K. Tang, S.H. Low, H. Hjalmarsson, An improved link model for window flow control and its application to FAST TCP, IEEE Trans. Automat. Contr. 54 (3) (2009) 551–564.

[16] K. Jacobsson, H. Hjalmarsson, N. Möller, ACK-clock dynamics in network congestion control – an inner feedback loop with implications on inelastic flow impact, in: Proc. IEEE CDC, December 2007, pp. 1882–1887.

[17] C. Jin, D. Wei, S.H. Low, FAST TCP: motivation, architecture, algorithms, performance, in: Proc. IEEE INFOCOM, March 2004, pp. 2490–2501.

[18] C. Jin, D.X. Wei, S.H. Low, G. Buhrmaster, J. Bunn, D.H. Choe, R.L.A. Cottrell, J.C. Doyle, W. Feng, O. Martin, H. Newman, F. Paganini, S. Ravot, S. Singh, FAST TCP: from theory to experiments, IEEE Netw. 19 (1) (2005) 4–11.

[19] R. Johari, D.K.H. Tan, End-to-end congestion control for the internet: delays and stability, IEEE/ACM Trans. Netw. 9 (6) (2001) 818–832.

[20] D. Katabi, M. Handley, C. Rohrs, Congestion control for high bandwidth delay product networks, in: Proc. ACM SIGCOMM, August 2002, pp. 89–102.

[21] T. Kelly, Scalable TCP: improving performance in high-speed wide area networks, Comput. Commun. Rev. 33 (2) (2003) 83–91.

[22] R. King, R. Riedi, R. Baraniuk, Evaluating and improving TCP-Africa: an adaptive and fair rapid increase rule for scalable TCP, in: Proc. PFLDnet, February 2005.

[23] D. Leith, R. Shorten, H-TCP protocol for high-speed long distance networks, in: Proc. PFLDnet, February 2004.

[24] X. Liu, Z. Guo, X. Wang, F. Chen, X. Lian, J. Tang, M. Wu, M.F. Kaashoek, Z. Zhang, D^3S: debugging deployed distributed systems, in: Proc. USENIX NSDI, April 2008, pp. 423–437.

[25] S. Mascolo, C. Casetti, M. Gerla, M.Y. Sanadidi, R. Wang, TCP Westwood: bandwidth estimation for enhanced transport over wireless links, in: Proc. ACM MOBICOM, 2001, pp. 287–297.

[26] V. Misra, W.-B. Gong, D. Towsley, A fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED, in: Proc. ACM SIGCOMM, August 2000, pp. 151–160.

[27] A.S. Tanenbaum, Computer Networks, Prentice Hall, 2002.

[28] A. Tang, K. Jacobsson, L.L.H. Andrew, S.H. Low, Linear stability analysis of FAST TCP using a new accurate link model, in: Proc. Annual Allerton Conference on Communication, Control, and Computing, 2006.

[29] J. Wang, A. Tang, S.H. Low, Local stability of FAST TCP, in: Proc. IEEE CDC, December 2004, pp. 1023–1028.

[30] J. Wang, D.X. Wei, S.H. Low, Modeling and stability of FAST TCP, in: Proc. IEEE INFOCOM, March 2005, pp. 938–948.

[31] R. Wang, K. Yamada, M.Y. Sanadidi, M. Gerla, TCP with sender-side intelligence to handle dynamic, large, leaky pipes, IEEE J. Select. Areas Commun. 23 (2) (2005) 235–248.

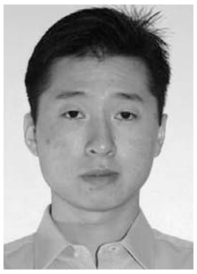
[32] Z. Wang, J. Crowcroft, A new congestion control scheme: slow start and search (Tri-S), ACM SIGCOMM Comput. Commun. Rev. 21 (1) (1991) 32–43.

[33] Z. Wang, J. Crowcroft, Eliminating periodic packet losses in the 43-Tahoe BSD TCP congestion control algorithm, ACM SIGCOMM Comput. Commun. Rev. 22 (2) (1992) 9–16.

[34] Y. Xia, L. Subramanian, I. Stoica, S. Kalyanaraman, One more bit is enough, in: Proc. ACM SIGCOMM, August 2005, pp. 37–48.

[35] L. Xu, K. Harfoush, I. Rhee, Binary increase congestion control (BIC) for fast, long distance networks, in: Proc. IEEE INFOCOM, March 2004, pp. 2514–2524.

[36] Y. Zhang, D. Leonard, D. Loguinov, JetMax: scalable max–min congestion control for high-speed heterogeneous networks, in: Proc. IEEE INFOCOM, April 2006, pp. 1–13.



Yueping Zhang received a B.S. degree in computer science from Beijing University of Aeronautics and Astronautics, Beijing, China, in 2001 and a Ph.D. degree in computer engineering at Texas A&M University, College Station, USA, in 2008. He is currently a Research Staff Member at NEC Laboratories America, Inc. His research interests include congestion control, delayed stability analysis, active queue management (AQM), router buffer sizing, and peer-to-peer networks.

Yong Xiong received the B.S. degree in electrical engineering from Tsinghua University, Beijing, China, in 1992, the M.S. degree in electrical engineering from Beijing Institute of Control Engineering, Chinese Academy of Space Technology, Beijing, China, in 1995, and the Ph.D. degree in computer science from Texas A&M University, College Station, TX, in 2005. He is currently with OPNET Technologies, Inc.

He was an Engineer with the Beijing Institute of Control Engineering from 1996 to 1998. His research interests are in the areas of Internet performance engineering including QoS, differentiated services, congestion control, and jitter control. He also works on control theory and digital signal processing.

Steve Liu received the B.S. and M.S. degrees in electrical engineering from National Cheng Kung University, Tainan, Taiwan, ROC, in 1979 and 1981, respectively. He received the Ph.D. degree from the University of Michigan, Ann Arbor, in 1989. Since then he has been a Senior Faculty Member with the Computer Science Department, Texas A&M University, College Station. He has worked on a broad range of basic and applied research problems, with over 60 published articles in professional journals, conferences, and workshops. His primary basic research interests are in the areas of real-time distributed systems, networking, and signal processing. In recent years, he has focused on issues related to bandwidth management solutions, signal processing techniques for analysis of freeway traffic, and Internet traffic and images. He is extensively engaged in interdisciplinary research such as intelligent transportation systems, telemedicine, and e-services, and he is a consultant for the high-tech industry.

Dmitri Loguinov received the B.S. degree (with honors) in computer science from Moscow State University, Russia, in 1995 and the Ph.D. degree in computer science from the City University of New York, New York, in 2002.

Between 2002 and 2007, he was an Assistant Professor in the Department of Computer Science at Texas A&M University, College Station. He is currently a tenured Associate Professor and Director of the Internet Research Lab (IRL) in the same department. His research interests include peer-to-peer networks, video streaming, congestion control, Internet measurement and modeling.