Towards Stability Analysis of Data Transport Mechanisms: a Fluid Model and Its Applications

Gayane Vardoyan, C.V. Hollot, and Don Towsley

***Abstract*—The Transmission Control Protocol (TCP) utilizes a congestion avoidance and control mechanism as a preventive measure against congestive collapse and as an adaptive measure in the presence of changing network conditions. The set of available congestion control algorithms is diverse, and while many have been studied from empirical and simulation perspectives, there is a notable lack of analytical work for some variants. To gain more insight into the dynamics of these algorithms, we: (1) propose a general modeling scheme consisting of a set of functional differential equations of retarded type (RFDEs) and of the congestion window as a function of time; (2) apply this scheme to TCP Reno and demonstrate its equivalence to a previous, well known model for TCP Reno; (3) show applications of the new framework to the widely-deployed congestion control algorithm TCP CUBIC, for which analytical models are few and limited; as well as to H-TCP, another high-speed congestion control algorithm; and (4) validate the model using simulations. Our modeling framework yields a fluid model for window- or rate-based congestion control variants. From a theoretical analysis of this model with TCP CUBIC, we discover that CUBIC is *locally uniformly asymptotically stable* – a property of the algorithm previously unknown. Through further analysis, we derive a sufficient condition for H-TCP's stability, but observe via a numerical analysis and simulations that H-TCP rarely converges to an equilibrium and is usually *not* asymptotically stable in practical high-speed settings.**

I. Introduction

TCP carries most of the traffic on the Internet. One of its important functions is to perform end-to-end congestion control to alleviate congestion in the Internet and to provide fair bandwidth sharing among different flows. To date, many different congestion control algorithms (variants) have been developed, among which are Reno, Vegas, STCP [1], CUBIC [2], H-TCP [3], and BBR [4]. Stability is an imperative property for any dynamical system. The stability of several of these variants including Reno, Vegas, and STCP has been extensively and carefully studied, however, little is known about the stability properties of more recent variants such as CUBIC and H-TCP. These latter variants have typically been studied through simulation and experimentation, neither of which are adequate to make careful statements about stability. As we will observe, for some variants this deficiency is due to the lack of a modeling framework with which to develop appropriate models that are amenable to a formal stability analysis. The goals of this paper are to point out deficiencies in the previous framework used to study variants such as Reno that make it unsuitable to study a variant such as CUBIC, and then to present a new framework and apply it to the analyses of CUBIC and H-TCP. Our choice of CUBIC is because it is a popular variant that is the default in the Linux distribution, and our choice of H-TCP is because it has been recommended by the Energy Sciences Network [5] and has been used within the Department of Energy's data transfer network [6].

The traditional approach for modeling a congestion control algorithm's behavior is to derive a differential equation (DE) for its congestion window (*cwnd*) or sending rate as a function of time. Such DEs typically account for the algorithm's increase and decrease rules, as well as loss probability functions, for example, to incorporate an active queue management (AQM) policy. This method is highly effective for modeling certain types of controllers, such as TCP Reno and STCP, whose *cwnd* update rules are very simple (*e.g.*, Reno's *cwnd* grows by one every round trip and decreases by half upon congestion detection). However, this approach reaches its limitations when presented with a controller whose *cwnd* update functions are complex, thereby making it difficult or impossible to directly write a DE for the *cwnd* or sending rate. For example, CUBIC's increase update rule is a function of *time since last loss* and of the *congestion window size immediately before loss*. Moreover, in the case of CUBIC, the steady-state value of *cwnd* lies at the saddle point of the window function, which obstructs the stability analysis of the protocol.

To overcome the impediments of the traditional approach, we develop a novel framework that exploits the fact that all *cwnd*- and rate-based controllers that utilize packet loss information[1] to make changes to the *cwnd* or rate have two variables in common: the value of *cwnd* (rate) immediately before loss and the time elapsed since last loss. As a consequence, one can derive a set of two DEs: the first describing the maximum *cwnd* (rate) as a function of time, and the second describing the duration of congestion epochs. This is a relatively easy task, compared to deriving a DE for *cwnd* (rate) of a complex algorithm directly. The advantage of such a model is that it offers tremendous versatility since it does not define *cwnd* or rate functions within the set of DEs, with the latter being identical for many controllers. Note that the proposed model is applicable not only to TCP-based congestion controllers, but also to UDT [7] and QCN [8].

In this work, we use both event- and packet-based simulations to validate our analytical models. To validate the analyt-

G. Vardoyan and D. Towsley are with the College of Information and Computer Sciences at the University of Massachusetts, Amherst.

C.V. Hollot is with the Department of Electrical and Computer Engineering at the University of Massachusetts, Amherst.

Manuscript received June 30, 2020.

[1]Note that this includes not only ACK-based algorithms, but also packet marking schemes as in ECN (Explicit Congestion Notification). From this point forward, we refer to such schemes as "loss-based".

ical stability results for CUBIC, we introduce a lightweight simulation framework that can easily be adapted to other congestion control variants. The simulation treats loss as a non-homogenous Poisson process and generates new loss events based on a user-defined loss model. We refer to this framework as the non-homogeneous Poisson loss (NHPL) simulation. The reason for its development is that it enables us to fully control every aspect of a model, such as changing link capacity throughout a flow's lifetime or using custom loss probability models. Perhaps the most important capability of this framework is the ease of specifying initial conditions for flows at the start of the congestion avoidance phase, since having control of these conditions is critical for testing the regions of stability for algorithms that are *locally* stable (as most congestion controllers in practice are). The NHPL simulation is of independent interest separate from this paper (a description of the framework is provided in [9] as well as in Appendix A).

Further, at the time of writing, NS3 [10] – a popular discrete-event network simulator – did not yet natively support CUBIC, and existing implementations have scalability problems: as more flows are introduced, the simulation becomes quite slow. Hence, we use the NHPL framework to validate the DE model for CUBIC and observe that the average *cwnd* predicted by both are in close agreement. As system parameters are varied, the simulation and CUBIC's DE model agree on whether the system is stable. For TCP CUBIC, we observe that instability can be introduced by setting the initial conditions too far from their fixed-point values. While our analysis states that CUBIC is locally asymptotically stable, these simulations complement the theory by demonstrating that CUBIC is *not* globally stable.

We use NS3, which unlike the NHPL framework is a packet-based simulation, to validate the DE model for H-TCP. We observe that NS3 is in close agreement with the DE model in terms of average *cwnd* and average congestion epoch duration. We also note that when H-TCP operates in its high-speed regime, its *cwnd* tends to exhibit large oscillations that last indefinitely, as evidenced by both the DE model and numerical evaluation of H-TCP's stability condition. In contrast, CUBIC's *cwnd* exhibits convergence to its fixed point, and fewer oscillations.

A summary of the contributions of this work is as follows:

- a new modeling framework applicable to a diverse set of congestion control algorithms,
- applications of this model to CUBIC and H-TCP, and stability analyses of these algorithms,
- validation of this model with two different simulation frameworks.

We call the new modeling framework the *MWLI (Max Window Loss Interval) model*. The MWLI model was originally presented in [11] with an extended version in [9]. Arguments for this model's validity (*e.g.*, a proof of concept with TCP Reno), as well as its use in proving the local asymptotic stability of CUBIC are covered in extensive detail in these manuscripts. Hence, we state only main results here for CUBIC's stability, but present some new analysis regarding its fixed point. For H-TCP, however, we present a stability analysis via the MWLI model in its entirety.

The rest of this paper is organized as follows: we discuss related work in Section II. We introduce the modeling framework in Section III and apply it to TCP Reno. In Section IV, we apply the MWLI model to TCP CUBIC and review results from its stability analysis from [11]. We also present a more detailed analysis of CUBIC's fixed point and observe its limiting behavior in terms of link capacity and delay. In Section V, we apply the model to H-TCP and present a sufficient condition for stability. Similar to the CUBIC analysis, we also perform a detailed analysis of H-TCP's fixed-point and derive its limiting behavior. In Section VI, we discuss the specific choice of a loss probability model used throughout our work. In Section VII, we validate the new model and the stability result for CUBIC and H-TCP using two different types of simulations and loss models. We draw conclusions in Section VIII.

II. Background

There exist a number of analytical studies of TCP and its stability. In [12], Misra *et al.* derive a fluid model for a set of TCP Reno flows and show an application to a networked setting where RED (Random Early Detection) is the AQM policy. Kelly proposed an optimization-based framework for studying and designing congestion control algorithms in [13], where STCP was an output. In [14], Srikant presented a simple analysis of Jacobson's TCP congestion control algorithm. In [15], Hollot *et al.* analyze the stability of TCP with an AQM system implementing RED.

Huang *et al.* develop and analyze the stability of a general nonlinear model of TCP in [16], focusing on HighSpeed, Scalable, and Standard TCP for comparisons of relative stability. The authors rely on functions $f(w)$ and $g(w)$, which are additive and multiplicative parameters, respectively, and are both functions of the current congestion window size. Our model differs from these examples in that rather than modeling the congestion window directly, we instead model two interdependent variables (maximum *cwnd* and time between losses) that in turn determine the evolution of the window. This new method presents a window of opportunity for modeling complex, nonlinear transport algorithms for which it is not possible to write a DE for *cwnd* directly or whose $f(w)$ and $g(w)$ functions cannot be written in closed form.

Theoretical analyses of TCP CUBIC are rare, possibly due to the protocol's behavior around the fixed point, which significantly complicates the analysis of its stability. In one study, Bao *et al.* propose Markov chain models for average steady-state TCP CUBIC throughput, in a wireless environment [17]. In [18], Poojary *et al.* derive an expression for average *cwnd* of a single CUBIC flow under random losses. In contrast to [17] and [18], the model we present in this work for CUBIC provides insight into both the transient and steady-state behavior of the algorithm. Moreover, we utilize Lyapunov stability theory to prove that CUBIC is locally asymptotically stable independent of link delay and other system parameters (the parameters only affect the region of attraction). This result is one of the main contributions of this work.

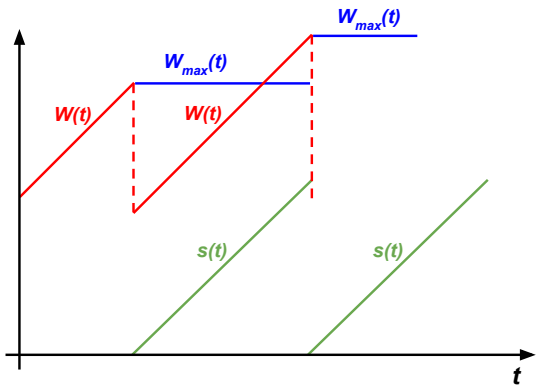


Fig. 1: $W(t)$, $W_{\max}(t)$, and $s(t)$ for TCP Reno.

Term	Definition
C	per-flow capacity
τ	link delay in signaling a loss to the source
$W_{\max}(t)$	the size of the *cwnd* immediately before loss
$s(t)$	the time elapsed since loss
$W(t)$	the *cwnd* as a function of time
$p(t)$	a probability of loss function

TABLE I: Term definitions.

Some studies attempt to gain insight into the stability of high-speed TCP variants using an empirical perspective. These usually involve characterizing stability (or instability) by the coefficient of variation (CoV) or stability index, as in [19] and [20], or even by simply using the standard deviation of the throughput as in [21]. While these observation-based and comparative studies are extremely valuable in assessing protocol behavior in deployment, they do not present a complete picture: to understand inherent protocol properties, modeling and performance analysis are required.

III. The MWLI Model

In this section, we present the new model, which is the focus of this work. As a proof of concept, we apply this model to TCP Reno and show that it is mathematically equivalent to the well-known DE model originally presented in [12]. We note that while the two models are equivalent, they make use of different types of information, which is essential for developing a fluid model for TCP CUBIC presented in Section IV and for H-TCP presented in Section V.

In the analysis that follows, we will use the notation $f \equiv f(t)$ to represent a function or variable that is not time-delayed. Similarly, we will use $f_T \equiv f(t-T)$ to represent a function or variable that is delayed by an amount of time T. We will also use $\dot{f} = df(t)/dt$ to represent the derivative of f with respect to time. The notation $\partial f(x)/\partial x$ denotes the partial derivative of f with respect to the variable x, and $\frac{\partial f(x)}{\partial x}|_{x=x^*}$ is the partial evaluated at $x = x^*$.

Table I presents some useful definitions. The main idea behind the model is the following: instead of deriving a DE for the *cwnd* function $W(t)$ directly, which is specific to a data transport algorithm, we instead derive DEs for $W_{\max}(t)$ – the size of the *cwnd* immediately before the most recent loss, and $s(t)$ – the amount of time elapsed since last loss, which are variables common to all loss-based algorithms. Since $W(t)$ is a function of $W_{\max}(t)$ and $s(t)$, it is completely determined by their DEs. The result is the following model[2]:

$$\begin{aligned} \frac{dW_{\max}(t)}{dt} &= -(W_{\max}(t) - W(t))\frac{W(t-\tau)}{\tau}p(t-\tau) \\ \frac{ds(t)}{dt} &= 1 - s(t)\frac{W(t-\tau)}{\tau}p(t-\tau) \end{aligned} \quad (1)$$

Here, $p(t-\tau)$ is a loss probability function. The expression $W(t-\tau)p(t-\tau)/\tau$ describes the packet loss rate, delayed by τ. Here, τ represents the delay in signaling a loss to the source, and may as well be a function of t. In the interest of simplifying notation (as well as subsequent analyses), we keep τ constant henceforth. Note that this assumption is reasonable for high-BDP regimes for which controllers like CUBIC and H-TCP were designed. The first DE in (1) describes the behavior of $W_{\max}$, which takes the value of $W(t)$ right before a loss. At the time of loss, if $W_{\max}(t) > W(t)$, then $W_{\max}$ decreases by the amount $W_{\max}(t) - W(t)$; otherwise, it increases by the same amount. The second DE describes the evolution of the time since last loss $s(t)$, which grows by one unit and is reset to zero upon loss. This system can be adapted to a rate-based scheme in terms of maximum rate and time since last rate decrease, simply by dividing each DE by τ. Since we will be describing applications of this model to TCP Reno, CUBIC, and H-TCP, which are all *cwnd*-based congestion controllers, we use (1) in the interest of the paper.

Figure 1 illustrates $W_{\max}(t)$, $s(t)$, and $W(t)$ for TCP Reno. To adapt model (1) to TCP Reno, we define Reno's *cwnd* as a function of $W_{\max}(t)$ and $s(t)$. At the time of loss, $W(t) = W_{\max}(t)$ is halved. This becomes the initial value of $W(t)$ in the new congestion epoch. $W(t)$ then increases by one segment for every round-trip time, so the total increase is $s(t)/\tau$ after $s(t)$ time has elapsed since the last loss. Hence,

$$W(t) = \frac{W_{\max}(t)}{2} + \frac{s(t)}{\tau}. \quad (2)$$

Then the fluid model for Reno is (1) combined with (2). In [11], we show that this model is mathematically equivalent to the well-established model for TCP Reno's *cwnd* presented in [12] (given by their equation (4) to be precise):

$$\frac{dW(t)}{dt} = \frac{1}{\tau} - \frac{W(t)}{2}\frac{W(t-\tau)}{\tau}p(t-\tau). \quad (3)$$

When used with Reno, the MWLI model given by (1) can be linearized. This representation can then be analyzed to yield system parameter-dependent conditions for Reno's stability. This analysis is similar to the one presented in [14].

The loss probability function can be customized according to the specific characteristics of a given system, such as queue size and AQM policy. For simplicity, when performing a formal stability analysis of a model, we use the following function:

$$p(t) = \max\left(1 - \frac{C\tau}{W(t)}, 0\right). \quad (4)$$

[2]Note that $W(t)$ must be either derived explicitly, for example as in (2) for TCP Reno or given in the definition of the controller, as in (5) for CUBIC.

This function is presented in [14] as an approximation of the M/M/1/B drop probability when the buffer size $B \to \infty$. We will explore an alternate loss probability model – one that incorporates more realistic queueing dynamics – when validating H-TCP using NS3.

Note that model (1) does not specify $W(t)$, and therein lies the versatility of this scheme. For a given *cwnd*-based transport algorithm, the modeler need only substitute a function describing the evolution of *cwnd* over time, as we did for Reno. We demonstrate this technique again with CUBIC in Section IV. This property of the model is useful both for analyzing existing algorithms and examining the stability of new ones. Moreover, the MWLI model can be used to help design and fine tune new congestion control algorithms, as their behavior may be simulated efficiently and easily using (1) and any loss probability function.

IV. Analysis of TCP CUBIC

In this section, we review the stability analysis of TCP CUBIC using the MWLI model, first presented in [11]. There, we showed that CUBIC is locally uniformly asymptotically stable using Razumikhin's Theorem (Theorem 1.4 in [22]) applied to a suitable Lyapunov function. Further, we derived convergence results on the system's solution. As a new contribution, we introduce a result regarding the limiting behavior of CUBIC's fixed point in terms of capacity and delay.

A. TCP CUBIC Fluid Model

TCP CUBIC's congestion window function is defined in terms of the time since last loss $s(t)$ and maximum value of *cwnd* immediately before the last loss $W_{\max}(t)$ [2]:

$$W(t) = c\left(s(t) - \sqrt[3]{\frac{W_{\max}(t)b}{c}}\right)^3 + W_{\max}(t) \qquad (5)$$

where b is a multiplicative decrease factor and c is a scaling factor. Figure 2 illustrates the evolution of CUBIC's *cwnd* over time. The opaque red curves represent behavior in steady state: the window is concave until a loss occurs at CUBIC's fixed-point value of *cwnd*, $\hat{W}$. The light red curves describe *cwnd* behavior if a loss does not occur: the window becomes convex, also known as CUBIC's probing phase. The fluid model for CUBIC is then simply (1) coupled with (5), with (4) as the loss probability function. More specifically, (1), (4), and (5) represent the dynamical interaction of CUBIC with a congested network. Prior to the development of (1), we attempted to develop a fluid model by first computing the equilibrium point for CUBIC, but this exercise gave a value of s at (5)'s saddle point and consequently, a confounding linearization of $dW/dt = 0$. Further attempts at deriving dW/dt, taking into account the time-dependencies $s(t)$ and $W_{\max}(t)$, resulted in a highly complex DE involving both $W_{\max}(t)$, $s(t)$, and their derivatives. Even obtaining the fixed points of this DE would be highly cumbersome, compared to obtaining the fixed point of (1).

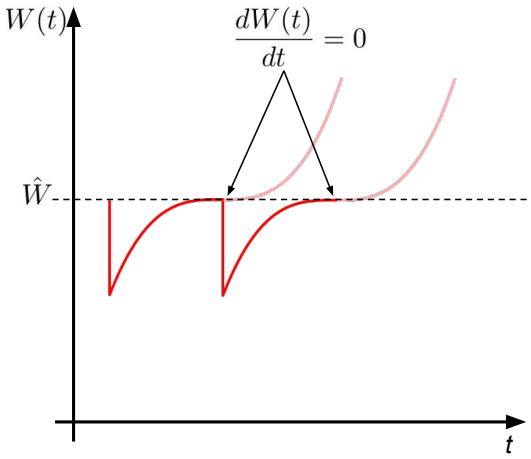


Fig. 2: CUBIC's saddle point causes $dW(t)/dt$ to evaluate to zero at the fixed point of the system.

B. Fixed Point Analysis

For (1), (4), and (5), let $\hat{W}_{\max}$, $\hat{s}$, $\hat{W}$, and $\hat{p}$ represent the fixed point values of $W_{\max}(t)$, $s(t)$, $W(t)$, and $p(t)$, respectively. In [11], we showed that

$$\hat{s} = \sqrt[3]{\frac{\hat{W}b}{c}} \quad \text{and} \quad \hat{W}(\hat{W} - C\tau)^3 = \frac{\tau^3 c}{b},$$

where the second equation can be solved for $\hat{W}$ solely as a function of the system parameters c, b, C, and τ. This value of $\hat{W}$ can then be used with the equation for $\hat{s}$ to obtain a value for $\hat{s}$ solely as a function of the system parameters. Further, we showed that $\hat{W}_{\max} = \hat{W}$ and that in steady state, $\hat{W}$ and $\hat{p}$ are strictly positive. Given the former fact, we use $\hat{W}_{\max}$ and $\hat{W}$ interchangeably from now on. An interesting comparison is $\hat{W}$ as a function of $\hat{p}$ for Reno and CUBIC. Model (3) yields

$$\hat{W}_{Reno} = \sqrt{\frac{2}{\hat{p}}}, \quad \text{while} \quad \hat{W}_{CUBIC} = \sqrt[4]{\frac{\tau^3 c}{\hat{p}^3 b}}.$$

In other words, whereas throughput under Reno depends on loss probability as $\mathcal{O}(\hat{p}^{-1/2})$, CUBIC exhibits a $\mathcal{O}(\hat{p}^{-3/4})$ dependence.

Next, we obtain the limiting behavior of $\hat{s}$.

Claim: $\hat{s} = \mathcal{O}((C\tau)^{1/3})$.

Proof: In [11], we showed that

$$\hat{s} = \frac{\tau}{\hat{W}\hat{p}} = \frac{\tau}{\hat{W} - C\tau}. \qquad (6)$$

Since $\hat{s} = (b\hat{W}/c)^{1/3}$, we also have that $\hat{W} = \hat{s}^3 c/b$. Substituting this expression for $\hat{W}$ into Eq. (6) and rearranging yields

$$\hat{s}^4 - \frac{b}{c}C\tau\hat{s} - \frac{b}{c}\tau = 0.$$

Given a quartic equation

$$c_4 x^4 + c_3 x^3 + c_2 x^2 + c_1 x + c_0 = 0,$$

the roots are

$$x_{1,2} = -\frac{c_3}{4c_4} - S \pm \frac{1}{2}\sqrt{-4S^2 - 2p + \frac{q}{S}},$$
$$x_{3,4} = -\frac{c_3}{4c_4} + S \pm \frac{1}{2}\sqrt{-4S^2 - 2p - \frac{q}{S}},$$

where,

$$p = \frac{8c_4c_2 - 3c_3^2}{8c_4^2}, \quad q = \frac{c_3^3 - 4c_4c_3c_2 + 8c_4^2c_1}{8c_4^3},$$
$$S = \frac{1}{2}\sqrt{-\frac{2}{3}p + \frac{1}{3c_4}\left(Q + \frac{\Delta_0}{Q}\right)}, \tag{7}$$
$$Q = \sqrt[3]{\frac{\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3}}{2}}, \tag{8}$$
$$\Delta_0 = c_2^2 - 3c_3c_1 + 12c_4c_0, \text{ and}$$
$$\Delta_1 = 2c_2^3 - 9c_3c_2c_1 + 27c_3^2c_0 + 27c_4c_1^2 - 72c_4c_2c_0.$$

For CUBIC, $c_3 = 0$, $c_4 = 1$, $p = 0$, $q = -C\tau\frac{b}{c}$, $\Delta_0 = -12b\tau/c$, and $\Delta_1 = 27C^2(b\tau/c)^2$, so that

$$Q = \sqrt[3]{\frac{27}{2}\left(\frac{b}{c}C\tau\right)^2 + \frac{1}{2}\sqrt{27^2\left(\frac{b}{c}C\tau\right)^4 + 4\left(12\frac{b}{c}\tau\right)^3}}$$
$$= \mathcal{O}((C\tau)^{2/3}),$$

and $$S = \frac{1}{2}\sqrt{\frac{1}{3}\left(Q - \frac{12b\tau}{c}\frac{1}{Q}\right)} = \mathcal{O}((C\tau)^{1/3}).$$

Finally, to obtain $\hat{s}$, we choose the root $x_{3,4}$ with the plus sign and show later on that this root is indeed valid (*i.e.*, positive for any large $C\tau$). Thus,

$$\hat{s} = S + \frac{1}{2}\sqrt{-4S^2 - \frac{q}{S}} = \mathcal{O}((C\tau)^{1/3}). \tag{9}$$

□

In contrast, for TCP Reno it can be shown that $\hat{s} = \mathcal{O}(C\tau^2)$.

Claim: for large $C\tau$, the quantity inside the square root in Eq. (9) is positive.

Proof: substituting the value for q above, we check if

$$\frac{C\tau b}{c}\frac{1}{S} - 4S^2 > 0. \tag{10}$$

When Eq. (10) holds, substituting for S and rearranging yield

$$\frac{2C\tau b}{c} > \left(\frac{1}{3}\left(Q - \frac{12b\tau}{c}\frac{1}{Q}\right)\right)^{3/2}.$$

Note that Q is always positive. Hence, it suffices to show

$$\frac{3^{3/2}2C\tau b}{c} > Q^{3/2}.$$

Substituting for Q and dividing by $\sqrt{27}$ yields

$$\frac{2C\tau b}{c} > \frac{1}{\sqrt{2}}\sqrt{\left(\frac{b}{c}C\tau\right)^2 + \sqrt{\left(\tau\frac{b}{c}\right)^3\left(C^4\tau\frac{b}{c} + \frac{256}{27}\right)}}$$

For large $C\tau$,

$$C^4\tau\frac{b}{c} + \frac{256}{27} < 2C^4\tau\frac{b}{c},$$

so it suffices to show

$$\frac{2C\tau b}{c} > \frac{1}{\sqrt{2}}\sqrt{\frac{b^2}{c^2}C^2\tau^2(1+\sqrt{2})}.$$

Noting that $1 + \sqrt{2} < 4$, it suffices to show that

$$\frac{2C\tau b}{c} > \sqrt{2\frac{b^2}{c^2}C^2\tau^2} = \sqrt{2}\frac{b}{c}C\tau,$$
$$2 > \sqrt{2}\ \checkmark$$

Hence, for large $C\tau$, we have selected the correct root for computing $\hat{s}$. □

C. Change of Variables

For convenience, we perform a change of variables within the MWLI model, (1), so that the fixed point of the system is located at the origin. To accomplish this, define $\mathbf{x}$ as follows:

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} W_{\max}(t) - \hat{W}_{\max} \\ s(t) - \hat{s} \end{bmatrix} = \begin{bmatrix} W_{\max}(t) - \hat{W} \\ s(t) - \hat{s} \end{bmatrix}$$

where the last equality follows because $\hat{W}_{\max} = \hat{W}$. Also, define $\Psi(t)$ and $\tilde{p}(t)$ as follows:

$$\Psi(t) = c\left(x_2(t) + \hat{s} - \sqrt[3]{\frac{b(x_1(t) + \hat{W})}{c}}\right)^3 + x_1(t) + \hat{W},$$
$$\tilde{p}(t) = \max\left(1 - \frac{C\tau}{\Psi(t)}, 0\right).$$

Then the new system is:

$$\begin{aligned} \dot{x}_1 &= \left(\Psi - x_1 - \hat{W}\right)\frac{\Psi_\tau}{\tau}\tilde{p}_\tau, \\ \dot{x}_2 &= 1 - (x_2 + \hat{s})\frac{\Psi_\tau}{\tau}\tilde{p}_\tau. \end{aligned} \tag{11}$$

Note that Ψ_τ and $\tilde{p}_\tau$ are functions of $x_1(t-\tau) \equiv x_{1_\tau}$ and $x_2(t-\tau) \equiv x_{2_\tau}$. It is easy to verify that $\mathbf{x}^* = [x_1\ x_2\ x_{1_\tau}\ x_{2_\tau}]^T = \mathbf{0}$ is a fixed point of the new system (see [11] for a proof). Stability analysis of the system (11) at the origin is equivalent to analyzing the stability of the original system (1) at the equilibrium values $\hat{W}_{\max}$ and $\hat{s}$.

D. Stability and Convergence Analyses

In [11], we show the existence and uniqueness of a solution to (11) through its fixed point $\mathbf{x}^*$. This is done by showing that $\dot{x}_1$ and $\dot{x}_2$ are continuously differentiable functions in some neighborhood of the fixed point, so that we have local Lipschitz continuity. Recall that in this work, we prove *local* stability for TCP CUBIC (there is evidence that the system is not globally stable, as we discuss in Section VII). Specifically, we prove that there exists a neighborhood around the fixed point such that the system is stable. While we do not provide a detailed characterization of the domain of attraction, we are able to show that x_1 and x_{1_τ} should be restricted to an interval $[-\rho\hat{W}, \rho\hat{W}]$, for some $\rho \in (0, 1)$.

Next, we cover the main points and results of the stability analysis. In general, the linearization of (11) and (4) about $\mathbf{x} = \mathbf{x}^* \coloneqq \mathbf{0}$ is

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \frac{1}{\hat{s}}A_0\begin{bmatrix}x_1\\ x_2\end{bmatrix} - \frac{\hat{s}}{\tau}A_1\begin{bmatrix}x_{1_\tau}\\ x_{2_\tau}\end{bmatrix}, \text{ where}$$

$$A_0 = \begin{bmatrix}\frac{\partial\Psi}{\partial x_1} - 1 & \frac{\partial\Psi}{\partial x_2}\\ 0 & -1\end{bmatrix}\Bigg|_{\mathbf{x}=\mathbf{x}^*} \text{ and } A_1 = \begin{bmatrix}0 & 0\\ \frac{\partial\Psi_\tau}{\partial x_{1_\tau}} & \frac{\partial\Psi_\tau}{\partial x_{2_\tau}}\end{bmatrix}\Bigg|_{\mathbf{x}=\mathbf{x}^*}.$$

For CUBIC, $\partial\Psi/\partial x_1|_{\mathbf{x}=\mathbf{x}^*} = 1$ and $\partial\Psi/\partial x_2|_{\mathbf{x}=\mathbf{x}^*} = 0$, so that $\dot{x}_1 = 0$. This means that the linearized system cannot be used to deduce the local stability of the nonlinear system. The key cause of this problem is the fact that the fixed point value of $x_2 = 0$ is the saddle point of the function Ψ (or equivalently, $\hat{s}$ is the saddle point of $W(t)$). This causes all first-order partial derivatives of $\dot{x}_1$ to evaluate to zero at $\mathbf{x}^* = \mathbf{0}$. Figure 2 illustrates this phenomenon. Hence, in order to incorporate a local contribution from $\dot{x}_1$ in the analysis, it is necessary to expand $\dot{x}_1$ further. Specifically, a third-order Taylor Series expansion is necessary, since all second-order terms also evaluate to zero at the origin.

The expanded system looks as follows:

$$\begin{aligned}\dot{x}_1 &= -\alpha x_1^3 + \beta x_1^2 x_2 - \gamma x_1 x_2^2 + \delta x_2^3 + h_1\\ \dot{x}_2 &= -\frac{1}{\hat{s}}x_2 - \frac{\hat{s}}{\tau}x_{1_\tau} + h_2\end{aligned} \tag{12}$$

where $\alpha = \dfrac{b^3}{27c^2\hat{s}^7},\ \beta = \dfrac{b^2}{3c\hat{s}^5},\ \gamma = \dfrac{b}{\hat{s}^3},$ and $\delta = \dfrac{c}{\hat{s}}$

and h_1 and h_2 are higher-order terms of $\dot{x}_1$ and $\dot{x}_2$, respectively. In [11], we analyze the stability of (12) using the Lyapunov-Razumikhin Theorem, the statement of which is given below as it appears in [22]. For the purpose of this theorem, we introduce some notation. Let $\mathcal{C} = \mathcal{C}([-\tau, 0], \mathbb{R}^n)$ be the set of continuous functions mapping the interval $[-\tau, 0]$ to $\mathbb{R}^n$, where τ is the maximum delay of a system. For any $A > 0$ and any continuous function of time $\psi \in \mathcal{C}([t_0 - \tau, t_0 + A], \mathbb{R}^n)$, and $t_0 \le t \le t_0 + A$, let $\psi_t \in \mathcal{C}$ be a segment of the function ψ defined as $\psi_t(\theta) = \psi(t + \theta)$, $-\tau \le \theta \le 0$. The general form of a retarded functional differential equation is

$$\dot{x}(t) = f(t, x_t) \tag{13}$$

Below, $\mathbb{R}_+$ is the set of positive real numbers, and $\bar{\mathbb{S}}$ is the closure of the set $\mathbb{S}$.

Theorem IV.1 (Lyapunov-Razumikhin Theorem)**.** *Suppose $f : \mathbb{R} \times \mathcal{C} \to \mathbb{R}^n$ takes $\mathbb{R}\times$(bounded sets of $\mathcal{C}$) into bounded sets of $\mathbb{R}^n$, and u, v, $w : \bar{\mathbb{R}}_+ \to \bar{\mathbb{R}}_+$ are continuous nondecreasing functions, $u(s)$ and $v(s)$ are positive for $s > 0$, and $u(0) = v(0) = 0$, v strictly increasing. If there exists a continuously differentiable function $V : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$ such that*

$$u(||x||) \le V(t, x) \le v(||x||), \quad \text{for } t \in \mathbb{R} \text{ and } x \in \mathbb{R}^n, \tag{14}$$

$w(s) > 0$ for $s > 0$, and there exists a continuous nondecreasing function $p(s) > s$ for $s > 0$ such that

$$\dot{V}(t, x(t)) \le -w(||x(t)||) \tag{15}$$

$$\text{if } V(t + \theta, x(t + \theta)) \le p(V(t, x(t))) \tag{16}$$

for $\theta \in [-\tau, 0]$, then the system (13) is uniformly asymptotically stable. If in addition $\lim_{s\to\infty} u(s) = \infty$, then the system (13) is globally uniformly asymptotically stable.

The continuous nondecreasing function $p(s)$ defined in the theorem above is not to be confused with the loss probability function defined for the MWLI model earlier in the text. Note that in this work, we will only prove *local* stability for CUBIC. Therefore, our goal is to show that we can find a function V for which all conditions specified in the theorem are valid locally, *i.e.*, in a sufficiently small neighborhood around the fixed point. A typical choice of Lyapunov candidate is the quadratic form, *i.e.*, a function of the form $Z(\mathbf{x}) = \mathbf{x}^T P \mathbf{x}$, where P is a positive definite matrix. In [11], we discuss in detail why such a candidate is not suitable for analyzing the stability of our system and instead propose the Lyapunov-Razumikhin candidate

$$V(\mathbf{x}) = \frac{\hat{s}}{2c}x_1^2 + \frac{\tau}{4\hat{s}}x_2^4. \tag{17}$$

We prove this candidate's validity (*i.e.*, it satisfies conditions (14), (15), and (16)) and use it to prove the local uniform asymptotic stability of TCP CUBIC around its fixed point $\mathbf{x}^* = \mathbf{0}$. Note that we can write $V(\mathbf{x})$ or $V(\mathbf{x}(t))$ instead of $V(t, \mathbf{x}(t))$ because V is autonomous, *i.e.*, it is not explicitly a function of time. Specifically, we showed that condition (14) is satisfied under

$$u(||\mathbf{x}||) = \epsilon_1||\mathbf{x}||_2^4 \quad \text{and} \quad v(||\mathbf{x}||) = \epsilon_0||\mathbf{x}||_2^2, \quad \text{where}$$
$$\epsilon_1 < \min\left(\frac{\hat{s}}{6c}, \frac{\tau}{4\hat{s}}\right) \quad \text{and} \quad \epsilon_0 = \max\left(\frac{\hat{s}}{2c}, \frac{\tau}{4\hat{s}}\right).$$

For condition (16), we let $p > 1$ be a constant, which can be arbitrarily close to one. Then for (16), we can use $p(V(\mathbf{x}(t))) = pV(\mathbf{x}(t))$:

$$V(\mathbf{x}(t - \theta)) \le pV(\mathbf{x}(t)), \text{ for } \theta \in [0, \tau].$$

Finally, for condition (15) we showed that (under condition (16)) there exists a constant $K < \lambda_{\min}[\tilde{Q}]$ such that

$$\dot{V} \le -(\lambda_{\min}[\tilde{Q}] - K)||\mathbf{x}||_2^4$$

where $\lambda_{\min}[\tilde{Q}]$ is the smallest eigenvalue of the matrix

$$\tilde{Q} = \begin{bmatrix}\frac{\hat{s}\alpha}{c} & -\frac{\hat{s}\beta}{2\sqrt{2}c} & 0\\ -\frac{\hat{s}\beta}{2\sqrt{2}c} & \frac{\hat{s}\gamma}{2c} & 0\\ 0 & 0 & \frac{\tau}{\hat{s}^2}\end{bmatrix}.$$

In addition, we also showed the following convergence result:

$$||\mathbf{x}||_2^4 \le \frac{1}{\frac{\epsilon_1(\lambda_{\min}[\tilde{Q}] - K)}{\epsilon_0^2}t + \frac{\epsilon_1}{V(0)}}. \tag{18}$$

Above, we define $V(0) \equiv V(\mathbf{x}(0))$ since V is implicitly a function of time. The result in (18) can be contrasted with linearizable, stable systems, which exhibit exponential convergence to the fixed point. What follows is a summary of the stability and convergence results for TCP CUBIC. For the system described by (11), the following properties hold:

(a) The system has a unique fixed point $\mathbf{x}^* = \mathbf{0}$.

(b) The system has a unique solution in a neighborhood of this fixed point.

(c) The fixed point is locally uniformly asymptotically stable and in addition, we have the following constraints on the stability region:

(i) x_1 and x_{1_τ} are constrained to $[-\rho\hat{W}, \rho\hat{W}]$, for some $\rho \in (0,1)$.

(ii) $|x_1|, |x_2| < 1$.

(d) The solution is bounded according to (18) for $|x_1|$ and $|x_2|$ small enough.

V. Analysis of H-TCP

In this section, we use the MWLI model to analyze the stability of H-TCP. As with CUBIC, we use (4) as the loss model. Unlike CUBIC, whose *cwnd* is given explicitly as a function of time, H-TCP's *cwnd* behavior is stated in terms of increase and decrease response functions, so we must derive its $W(t)$ in closed form. H-TCP behaves as follows [23]: on each acknowledgement, set

$$cwnd \leftarrow cwnd + \alpha(\Delta)/cwnd \tag{19}$$

and on each congestion event, set

$$cwnd \leftarrow \beta(t) cwnd, \tag{20}$$

where Δ is the time since the last congestion event and $\alpha(\Delta)$ and $\beta(t)$ are described in detail below. $\alpha(\Delta)$ is constructed so that after a congestion event, H-TCP's *cwnd* increases linearly for a period of Δ^L seconds; this is known as the low-speed regime of the protocol. After Δ^L seconds have passed, as long as there are no new congestion events, H-TCP transitions into its high-speed regime and evolves the *cwnd* according to a function of Δ. When a congestion event occurs, H-TCP uses a backoff factor $\beta(t) \in [0.5, 0.8]$, depending on current estimated maximum and minimum round-trip times. The full operation is as follows:

(a) On each acknowledgement, set

$$\overline{\alpha}(\Delta) = \begin{cases} 1, & \text{if } \Delta \le \Delta^L, \\ 1 + 10(\Delta - \Delta^L) + \left(\frac{\Delta - \Delta^L}{2}\right)^2, & \text{if } \Delta > \Delta^L, \end{cases}$$

and then let $\alpha(\Delta) = \max\left(2(1-\beta(t))\overline{\alpha}(\Delta), 1\right)$. (Note: the original definitions, *e.g.*, in [3], [23], simply state $\alpha(\Delta) = 2(1-\beta(t))\overline{\alpha}(\Delta)$. However, if $\Delta \le \Delta^L$, then $2(1-\beta(t))\overline{\alpha}(\Delta) = 2(1-\beta(t))$, which will be less than one for any value of $\beta(t) \in [0.5, 0.8]$ that is not 0.5. Since the increase factor must be at least one, we assume that $\alpha(\Delta) = 1$ in this case, hence the use of $\max$ above (also used in [24]).) Then, use Eq. (19) to adjust the *cwnd*.

(b) On each congestion event, set

$$\beta(t) = \frac{\tau_{\min}}{\tau_{\max}}, \quad \beta(t) \in [0.5, 0.8].$$

Then, use Eq. (20) to adjust the *cwnd*.

Above, $\tau_{\min}/\tau_{\max}$ is the ratio of minimum to maximum round-trip times experienced by the source. Δ^L is usually set to $1s$ [23]. The authors in [23] also suggest scaling $\alpha(\Delta)$ with round-trip time to make the increase rate and convergence time invariant with RTT, as well as to potentially reduce unfairness. We first derive $W(t)$ without RTT scaling and later on introduce a scaling factor γ to discuss its effects on stability ($\gamma = \tau/\tau_{ref}$, where τ_{ref} is a reference RTT value as discussed in [24] and [25]). To simplify our analysis, we make the assumption that $\beta(t) = b$, for $b \in [0.5, 0.8]$. Note that this assumption is reasonable, since in steady state, $\tau_{\min} = \tau_{\max}$ so that the decrease factor is a constant.

Note that in the case where $\Delta > \Delta^L$, $2(1-b)\overline{\alpha}(\Delta)$ can also be less than one, so that $\alpha(\Delta) = 1$. This means that until some time t_L, H-TCP will operate in the low-speed regime (*i.e.*, it will behave like TCP Reno even though $\Delta > \Delta^L$). We compute t_L to simplify future analysis. To do so, we solve the following equation for t_L:

$$2(1-b)\left(1 + 10(t_L - \Delta^L) + \left(\frac{t_L - \Delta^L}{2}\right)^2\right) = 1.$$

The result is

$$t_L = \Delta^L - 20 + \sqrt{400 - 4 + \frac{2}{(1-b)}}.$$

Note that for $b \in [0.5, 0.8]$, $t_L \ge \Delta^L$. We are now ready to write down the *cwnd* function for H-TCP:

$$W(t) = bW_{\max}(t) + \frac{1}{\tau}\min\left(t_L, s(t)\right) + \frac{\mathbb{1}\{s(t) > t_L\}}{\tau}\int_{t_L}^{s(t)} \alpha(\Delta) d\Delta. \tag{21}$$

Note that the indicator function above is required since the integral can evaluate to a positive value in some cases where $s(t) < t_L$.

A. Fixed Point Analysis

If $t_L \ge \hat{s}$, then the fixed point of the system and its stability analysis reduces to that of TCP Reno. From now on, we assume that $t_L < \hat{s}$; in other words, we analyze H-TCP in environments where the congestion epochs in steady state are sufficiently long for the *cwnd* to transition to the high-speed regime after the linear growth stage. This is common in high-BDP environments for which H-TCP was designed. Note that this causes the indicator function in (21) to evaluate to 1. In steady state, we know that $W(t) = W_{\max}(t) = \hat{W}$ and $s(t) = \hat{s}$. Applying this to (21), we have

$$\hat{W} = \frac{t_L}{(1-b)\tau} + \frac{2}{\tau}\left(\frac{(\hat{s} - \Delta^L)^3}{12} + 5(\hat{s} - \Delta^L)^2 + \hat{s} - t_L - \frac{(t_L - \Delta^L)^3}{12} - 5(t_L - \Delta^L)^2\right).$$

We now eliminate $\hat{W}$ from the equation above. From the second equation in (1), we obtain

$$\frac{\hat{s}\hat{W}\hat{p}}{\tau} = 1,$$

$$\frac{\hat{s}\hat{W}(1 - C\tau/\hat{W})}{\tau} = 1,$$

$$\hat{W} = \frac{\tau}{\hat{s}} + C\tau. \tag{22}$$

Substituting this into the fixed point equation above, we obtain

$$\frac{\tau}{\hat{s}} + C\tau = \frac{t_L}{(1-b)\tau} + \frac{2}{\tau}\left(\frac{(\hat{s}-\Delta^L)^3}{12} + 5(\hat{s}-\Delta^L)^2 + \hat{s} - t_L - \frac{(t_L-\Delta^L)^3}{12} - 5(t_L-\Delta^L)^2\right). \quad (23)$$

Note that (23) is quartic in $\hat{s}$. We can use it to analyze the limiting behavior of $\hat{s}$ in terms of C and τ (specifically, we assume large $C\tau$ and $C \gg \tau$). To simplify notation, let $x \equiv \hat{s}$. Recall that in steady state, $\tau_{\min} = \tau_{\max}$, which means that the decrease factor is $b = 0.8$. From now on, to simplify subsequent analysis, we assume $\Delta^L = 1$s, which is the usual default value in H-TCP implementations. Using this, along with the steady-state value of b, we may now evaluate the steady-state value of t_L, which is given by $\sqrt{406} - 19$. Substituting this into (23), expanding and grouping by x yields

$$x^4 + 57x^3 - 105x^2 + \left(812\sqrt{406} - 16283 - 6C\tau^2\right)x - 6\tau^2 = 0.$$

Recall the form of the roots of a quartic equation from Section IV-B. For H-TCP, letting $\Gamma = 812\sqrt{406} - 16283$, we have

$$\begin{aligned}
\Delta_0 &= 105^2 - 171\left(\Gamma - 6C\tau^2\right) - 72\tau^2 = \mathcal{O}(C\tau^2),\\
\Delta_1 &= -2(105^3) + 53865\left(\Gamma - 6C\tau^2\right) - 162(57^2)\tau^2\\
&\quad + 27\left(\Gamma - 6C\tau^2\right)^2 - 432(105)\tau^2 = \mathcal{O}(C^2\tau^4),\\
q &= \frac{1}{8}\left(57^3 + 228(105)\right) + \left(\Gamma - 6C\tau^2\right) = \mathcal{O}(C\tau^2),
\end{aligned}$$

and p is a negative constant (no dependence on C or τ). From above and from Eq. (8), we obtain

$$\Delta_1^2 = \mathcal{O}(C^4\tau^8),\ \Delta_0^3 = \mathcal{O}(C^3\tau^6),\ \text{and}\ Q = \mathcal{O}((C^2\tau^4)^{1/3}).$$

Further (see Appendix B for derivation details),

$$\frac{\Delta_0}{Q} = \mathcal{O}((C^2\tau^4)^{1/3}), \quad Q + \frac{\Delta_0}{Q} = \mathcal{O}((C^2\tau^4)^{1/3}), \quad \text{and}$$
$$S = \frac{1}{2}\sqrt{\frac{2}{3}p' + \frac{1}{3}\left(Q + \frac{\Delta_0}{Q}\right)} = \mathcal{O}((C\tau^2)^{1/3}),$$

where $p' \equiv -p$ is positive. Finally, letting $q' \equiv -q$ and noting that for large $C\tau$, $q' > 0$, we use the plus-sign version of $x_{3,4}$ to obtain

$$x = -\frac{57}{4} + S + \frac{1}{2}\sqrt{2p' + \frac{q'}{S} - 4S^2} \quad (24)$$

and further noting that q'/S and S^2 are both $\mathcal{O}((C^2\tau^4)^{1/3})$, we obtain $\hat{s} = \mathcal{O}((C\tau^2)^{1/3})$. See Appendix B for a proof of the limiting behavior of q'/S, as well as for a proof that the root in (24) is positive and real for large $C\tau$ and $C \gg \tau$.

B. Stability Analysis

We perform a change of variables similar to Section IV-C. Letting $\delta \equiv \hat{s} - \Delta^L$, $\xi \equiv t_L - \Delta^L$ and rewriting the *cwnd* function in terms of $x_1(t)$ and $x_2(t)$, yields

$$\Psi(t) = b(x_1(t) + \hat{W}) + \frac{t_L}{\tau} + \frac{2(1-b)}{\tau}\left(\frac{(x_2(t)+\delta)^3}{12} + 5(x_2(t)+\delta)^2 + x_2(t) + \hat{s} - t_L - \frac{\xi^3}{12} - 5\xi^2\right).$$

For the stability analysis, we need the partial derivatives of $\Psi(t)$ with respect to x_1 and x_2 evaluated at the fixed point. They are

$$\begin{aligned}
\frac{\partial\Psi}{\partial x_1} &= b,\\
\frac{\partial\Psi}{\partial x_2} &= \frac{2(1-b)}{\tau}\left(\frac{(x_2(t)+\delta)^2}{4} + 10(x_2(t)+\delta) + 1\right),\\
\left.\frac{\partial\Psi}{\partial x_2}\right|_{\mathbf{0}} &= \frac{2(1-b)}{\tau}\left(\frac{(\hat{s}-\Delta^L)^2}{4} + 10(\hat{s}-\Delta^L) + 1\right).
\end{aligned}$$

Let the quantity above (the partial with respect to x_2 evaluated at the fixed point) be denoted as F_{Δ^L}. Note that F_{Δ^L} is always positive: recall that $t_L \geq \Delta^L$ for all $b \in [0.5, 0.8]$ and by our assumption, $\hat{s} > t_L$, so it follows that $\hat{s} > \Delta^L$. The linearized system is

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \frac{1}{\hat{s}}\begin{bmatrix}b-1 & F_{\Delta^L}\\ 0 & -1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} - \frac{\hat{s}}{\tau}\begin{bmatrix}0 & 0\\ b & F_{\Delta^L}\end{bmatrix}\begin{bmatrix}x_{1_\tau}\\ x_{2_\tau}\end{bmatrix}.$$

Then, calling the first matrix (multiplied by $1/\hat{s}$) above A_0 and the second matrix (multiplied by $-\hat{s}/\tau$) A_1, note that the linearized system has the form

$$\dot{\mathbf{x}} = A_0\mathbf{x} + A_1\mathbf{x}_\tau.$$

Its Laplace transform is

$$s\mathbf{X}(s) - \mathbf{x}(0) = A_0\mathbf{X}(s) + A_1\mathbf{X}(s)e^{-s\tau}.$$

Solving for $\mathbf{X}(s)$ yields

$$\mathbf{X}(s) = (sI - A_0 - A_1e^{-s\tau})\mathbf{x}(0).$$

We compute

$$sI - A_0 - A_1e^{-s\tau} = \begin{bmatrix}s + \frac{1-b}{\hat{s}} & \frac{-F_{\Delta^L}}{\hat{s}}\\ \frac{b\hat{s}}{\tau}e^{-s\tau} & s + \frac{1}{\hat{s}} + \frac{\hat{s}}{\tau}F_{\Delta^L}e^{-s\tau}\end{bmatrix},$$

and the determinant of this matrix is

$$\Delta = \left(s + \frac{1-b}{\hat{s}}\right)\left(s + \frac{1}{\hat{s}}\right)\left(1 + \frac{\hat{s}}{\tau}\frac{F_{\Delta^L}}{\left(s + \frac{1-b}{\hat{s}}\right)}e^{-s\tau}\right).$$

Note that the first two components of Δ, $s + (1-b)/\hat{s}$ and $s + 1/\hat{s}$, are stable. For the third component, we can apply the Nyquist stability criterion. Let

$$H(s) \coloneqq \frac{\hat{s}}{\tau}\frac{F_{\Delta^L}}{\left(s + \frac{1-b}{\hat{s}}\right)}e^{-s\tau} = \frac{\hat{s}F_{\Delta^L}}{\left(s\tau + \frac{(1-b)\tau}{\hat{s}}\right)}e^{-s\tau}$$

and let $\rho \coloneqq (1-b)\tau/\hat{s}$ and $\theta \coloneqq \tan^{-1}(\omega\tau/\rho)$. Then

$$H(j\omega) = \frac{\hat{s}F_{\Delta^L}}{\sqrt{\omega^2\tau^2 + \rho^2}}e^{-j(\omega\tau+\theta)}.$$

Decomposing $H(j\omega)$ into real and imaginary components yields

$$\mathcal{I}(H(j\omega)) = -\frac{\hat{s}F_{\Delta^L}\sin(\omega\tau+\theta)}{\sqrt{\omega^2\tau^2 + \rho^2}},$$

$$\mathcal{R}(H(j\omega)) = \frac{\hat{s}F_{\Delta^L}\cos(\omega\tau+\theta)}{\sqrt{\omega^2\tau^2+\rho^2}}.$$

The Nyquist stability criterion states the following: *The rational function $1+H(s)$ has poles in the open left-half s-plane if and only if the Nyquist contour Γ_H in the $H(s)$-plane does not encircle the $(-1,0)$ point when the number of poles of $H(s)$ in the right-hand s-plane is zero* [26]. Clearly, $H(s)$ does not have any poles in the right-hand s-plane. It remains to check whether the point $(-1,0)$ is encircled in the $H(s)$-plane or to derive conditions to ensure that it is not encircled. We can find the point where the contour intersects the real axis by setting $\mathcal{I}(H(j\omega)) = 0$, yielding

$$\sin(\omega\tau+\theta) = 0.$$

This equation is satisfied when $\omega\tau+\theta = \pm n\pi$, for $n = 0,1,2,\ldots$. Next, we analyze the real component. For $n=0$ and n even, the numerator of the real part is equal to $\hat{s}F_{\Delta^L}$, which is always greater than zero. Hence, when $n=0$ or even, $\mathcal{I}(H(j\omega)) > 0$. To analyze the case of n odd, we use the following:

$$\omega\tau = \pm n\pi - \theta,$$
$$|\omega\tau| = |\pm n\pi - \theta|.$$

Further, note that $|\theta| = |\arctan(\omega\tau/\rho)| < \pi/2$. Therefore, $|\omega\tau| > \pi/2$, so for n odd,

$$\mathcal{R}(H(j\omega)) = \frac{-\hat{s}F_{\Delta^L}}{(>\pi/2)} > \frac{-2\hat{s}F_{\Delta^L}}{\pi}.$$

For stability, we require that $\mathcal{R}(H(j\omega)) \geq -1$. This yields the following (sufficient) stability condition:

$$\hat{s}F_{\Delta^L} \leq \frac{\pi}{2},$$
$$\hat{s}\left(\frac{(\hat{s}-\Delta^L)^2}{4} + 10(\hat{s}-\Delta^L)+1\right) \leq \frac{\pi\tau}{4(1-b)}.$$

Recall that $\hat{s}$ is a function of C, τ, and t_L, and the latter is a function of Δ^L. We have not been able to find values of C, τ, and Δ^L that satisfy the stability condition above. In addition, we have not been able to find a set of parameters for which the simulation of DEs show a stable system. Recall that in the *cwnd* function definition (21), no RTT scaling is employed. RTT scaling can reduce unfairness, as well as enable congestion epoch duration and convergence time to be independent of RTT. When implemented in H-TCP, RTT scaling is only enabled in high-speed regimes (*i.e.*, when $\Delta > t_L$), and works by scaling $\alpha(\Delta)$ in this regime by τ/τ_{ref}, where the recommended value of $\tau_{ref} = 100$ ms [24]. With this change, the stability condition becomes

$$\frac{\hat{s}}{\tau_{ref}}\left(\frac{(\hat{s}-\Delta^L)^2}{4} + 10(\hat{s}-\Delta^L)+1\right) \leq \frac{\pi}{4(1-b)},$$

where t_L is redefined as

$$t_L = \begin{cases} \Delta^L - 20 + \sqrt{400 - 4 + \frac{2\tau_{ref}}{(1-b)\tau}} & \text{if } \frac{2(1-b)\tau}{\tau_{ref}} \leq 1, \\ \Delta^L \text{ otherwise.} \end{cases}$$

Numerical analysis shows that in practical settings (*i.e.*, for realistic values of C, τ, and b), H-TCP with RTT scaling (with scaling parameter $\gamma := \tau/\tau_{ref}$) is still unstable when it operates in the high-speed regime. In fact, we find that the only way to stabilize the systems is to use values of b and τ_{ref} that force the protocol to operate exclusively in the low-speed regime (*i.e.*, to behave as standard TCP). However, we note that certain values of γ tend to dampen the oscillation amplitudes of H-TCP's *cwnd*, albeit without affecting the convergence behavior of the protocol. One example of this phenomenon is illustrated in Figure 3. The *cwnd* curves are a result of the DE model with $C = 1$ Gbps, $\tau = 40$ ms, and $b = 0.8$. $\gamma = 1$ corresponds to no RTT scaling, while $\gamma = 0.4$ corresponds to using the recommended $\tau_{ref} = 100$ ms. Recall the fixed point expression for *cwnd* from Eq. (22): $\hat{W} = \tau/\hat{s} + C\tau$. Note from this equation that in high-BDP settings, $\hat{W}$ is dominated by the BDP, $C\tau$, while varying γ only changes $\hat{s}$. Hence, $\hat{W}$ does not change much with γ. This is reflected in Figure 3, as all three flows reach approximately the same maximum value of *cwnd*, $\approx C\tau = 5\times10^6$ bytes. On the other hand, the flow that corresponds to $\gamma = 0.08$ deviates significantly less from the fixed point than do the other flows, with larger γ's. Interestingly, decreasing γ further precludes H-TCP from operating in the high-speed regime, and the protocol's behavior approaches that of standard TCP while not necessarily stabilizing it (*i.e.*, the *cwnd* retains its sawtooth profile and there is no convergence to the fixed point).

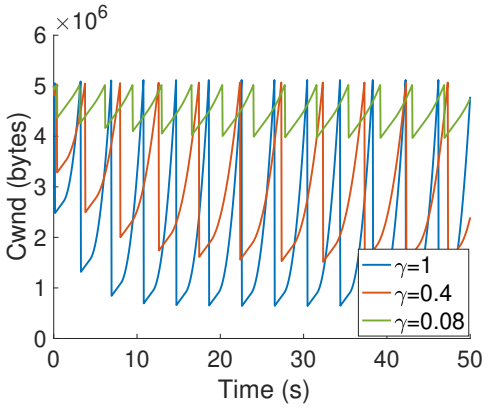


Fig. 3: Effect of RTT scaling on H-TCP's *cwnd*. Curves produced by the DE model with $C = 1$ Gbps, $\tau = 40$ ms, $b = 0.8$, and scaling factor $\gamma = \tau/\tau_{ref}$. In all three cases, the protocol operates in the high-speed regime and is unstable. Decreasing γ reduces the magnitude of the oscillations.

VI. A Note on the Loss Model and Other Applications of the MWLI Model

We chose the probability of loss model given by Eq. (4) for its simplicity and dependence on few system parameters. However, it is worthwhile to note that this loss model is used often in fluid approximation models, especially under the assumption that the packet arrival process is a Poisson process [14]. Such an assumption is reasonable in certain scenarios, *e.g.*, with sufficiently long-lived data transfers and large buffer sizes at congestion points. However, in scenarios where traffic may be bursty, the arrival process no longer behaves as a Poisson process, and using the loss model given by (4), or

even its finite-version for $M/M/1/B$ queues with queue size B, may cause inaccuracies for throughput prediction [27].

Note that the MWLI model can accommodate different loss models, and as long as a model $p(t)$ can be written in closed form in steady state (*e.g.*, (4) simply becomes $p(t) = 1 - C\tau/\hat{W}$ in steady state), one can perform fixed point and stability analyses. A consequence of using a particular $p(t)$ is that the analysis is valid under the same assumptions for which $p(t)$ is a reasonable model. Hence, while H-TCP seems unstable under (4), it may behave very differently under a less harsh $p(t)$. In Section VII, we explore another loss model for H-TCP that incorporates queue size.

Both CUBIC and H-TCP are second-generation TCP variants that were developed to operate more efficiently in high-BDP settings than their predecessors. In other words, these two protocols, which are both congestion-window based, fall into the same congestion controller category, and in our analyses we make some assumptions that are specific to TCP flows in high-BDP settings to enable simpler analyses. Nevertheless, it is possible to use the MWLI model to accommodate protocols and settings which we do not study in the present manuscript. For instance, the constant τ assumption may be removed, and τ replaced with a function $R(t)$ that depends explicitly on time, and implicitly on queueing dynamics as well as processing delay, the latter being an important factor in datacenter networks. In the next section, we demonstrate how to incorporate queueing dynamics $q(t)$ into a loss probability function $p(t)$; we also redefine τ as solely the propagation delay (which we assume to be constant), and define the overall delay to be the sum of queueing and propagation delays, much as in [28]. Making such modifications can render the model more predictive in settings where RTT variation is significant.

Further, there are some settings (*e.g.*, datacenters) in which buffers may be shallow. Such a scenario may be accommodated by choosing an appropriate queueing model with a maximum queue size q_{max} and adjusting this parameter accordingly. A similar change may be made for settings with large buffer sizes. Alternate loss probability models are also straightforward to implement, via the $p(t)$ function. For instance, to model TCP in a wireless setting, one would choose a $p(t)$ that reflects the completely independent Poisson-driven losses that are characteristic of such environments. To model bursty traffic one may use an on-off Markov process modulating a Poisson loss process. To account for ECN marks, one must account in the model that the rate at which congestion signals are relayed to the sender is a function of the queue length, which can also be done via an appropriately-chosen $p(t-\tau)$ function (which in turn may also have to reflect an AQM scheme). Finally, to account for duplicate ACKs, one may again use the $p(t-\tau)$ function to ensure that a loss is suffered when the buffer is full, so that a multiplicative decrease is applied to the *cwnd* as a result. We leave the details of these explorations to future work on the topic.

VII. Simulations

We use simulation to validate model (11) and the stability analyses of TCP CUBIC and H-TCP. For H-TCP, we use NS3 and for CUBIC, we develop a simulation framework that treats loss as a non-homogenous Poisson process and generates new loss events based on a user-defined probability of loss model. Because of the loss generation method, we call this simulation framework the Non-Homogeneous Poisson Loss, or NHPL simulation. A detailed description of the framework is provided in [9]. An advantage of using this framework for validating the DEs over, for example, NS3, is that we can observe the behavior of solely the congestion avoidance phase of an algorithm, which allows us to more easily verify the theoretical analysis of the controller's stability. Moreover, as we observe from simulations of the DEs, an algorithm's stability can be highly sensitive to the initial conditions specified at the beginning of the congestion avoidance phase (note that this observed sensitivity to initial conditions is consistent with our demonstration of local-only stability). The initial conditions are values of $W_{\max}(0)$ and $s(0)$ for all flows, and we can control them more easily with our simulation framework. This can be especially useful when testing the region of stability for a given system. Another reason for using NHPL simulations for CUBIC is that at the time of writing, CUBIC is not natively supported in NS3. As a result, certain experiments of CUBIC in NS3 take an exceedingly long time to complete (especially systems with high BDPs, which are of interest in this paper).

Note that the NHPL simulation framework is event-based, rather than packet-based. To diversify our experiments, we use NS3 – a packet-based simulation framework – to validate the H-TCP model. For the experimental setup, we use a `PointToPoint` channel between two net devices. We choose the PI (Proportional Integral controller) AQM scheme [29] for H-TCP in NS3. We choose PI because it is simple to implement within the MWLI model. Since NS3 does not natively support PI, we modify the similar AQM protocol PIE [30] implementation, which is available in NS3.

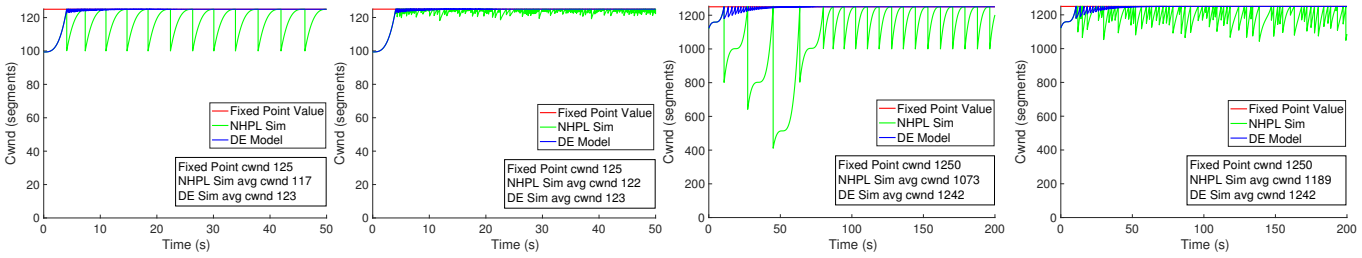
In order to compare the DE model against NS3, we alter the DEs' previous loss model (that of Eq. (4)) to incorporate some of the major components of PI, with the objective of implementing a simplified version of the AQM scheme that reasonably approximates PI. Specifically, every T_UPDATE seconds (we use 5 ms in all our experiments – similar to the value used in the experiments of [29]), we compute

$$\begin{aligned} &\Delta p = \hat{\alpha}(q - q_{ref}) + \hat{\beta}(q_{ref} - q_{old}) \quad \text{and set} \\ &p \leftarrow p + \Delta p \end{aligned} \tag{25}$$

as PI's new drop probability. Above, $\hat{\alpha}$ and $\hat{\beta}$ are parameters, q is the current length of the queue, q_{old} is the length of the queue in the previous iteration, and q_{ref} is the desired queue length. Note that this value of q_{ref} is a reference for an average flow within our DE model, as opposed to a reference for all flows in aggregate, and similarly for the values of q and q_{old}, so that the loss probability is computed for a single flow. In all our experiments, we set $\hat{\alpha} = 1.822 \times 10^{-5}$ and $\hat{\beta} = 1.816 \times 10^{-5}$, as in [29]. The value of p is then used to update s and $W_{\max}$ in the DE model.

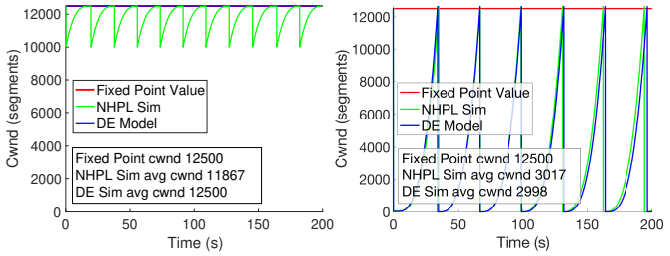
For the DE model, we use the following queue model (based on a model from [31]):

$$\frac{dq(t)}{dt} = \frac{W(t)}{\tau} - C, \tag{26}$$



(a) $\tau = 1$ ms, 1 flow (b) $\tau = 1$ ms, 20 flows (c) $\tau = 10$ ms, 1 flow (d) $\tau = 10$ ms, 20 flows

Fig. 4: Comparison of average *cwnd* (computed post-transient phase) generated by NHPL simulations against steady-state *cwnd* generated by model (11) for TCP CUBIC. Also shown is the fixed-point value of *cwnd*. Per-flow capacity $C = 1$ Gbps.



(a) $\tau = 100$ ms, stable (b) $\tau = 100$ ms, unstable

Fig. 5: The impact of initial conditions on stability. For both (a) and (b), $C = 1$ Gbps, $\tau = 100$ ms. In (a), there is one flow whose initial conditions $W(0)$ and $s(0)$ are very close to the fixed point values $\hat{W}$ and $\hat{s}$, respectively. Both the NHPL simulation and the model exhibit stability. In (b), there is one flow whose initial conditions are set too far from the fixed point value, destabilizing the flow in both the NHPL simulation and the DE system.

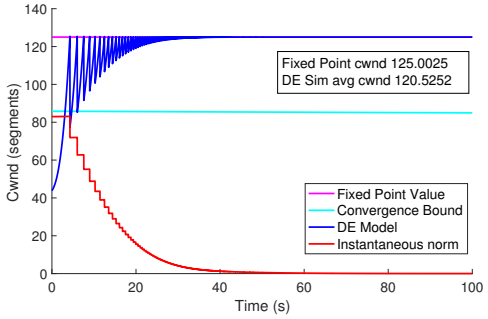


Fig. 6: Convergence for CUBIC. At the top is the *cwnd* generated by DEs as it converges to the fixed point value of *cwnd*. Below these two curves is a comparison of the instantaneous norm $||\mathbf{x}||_2$ against the analytical bound in (18). Here, $C = 100$ Mbps, $\tau = 10$ ms.

where $q(t)$ is the number of packets in the bottleneck queue. Then the queueing delay is given by $q_{del}(t) = q(t)/C$. Hence, the total delay is the sum of queueing and propagation delays, given by $q_{del}(t) + \tau$; this is the value we use instead of τ in (1). Note that for the experiment descriptions in this section, we only state the (constant) propagation delay τ.

It remains to choose a queue size for NS3 experiments: *i.e.*, the `MaxSize` variable for PI's queue, which we will call $q_{\max}$ from now on. Let N be the number of NS3 flows. For each NS3 experiment, we choose a value of N large enough so that bandwidth on the link is fully utilized. To choose a suitable value for $q_{\max}$, we multiply q_{ref} by N and choose a larger value, keeping in mind that choosing too large a $q_{\max}$ may cause instability.

For convenience, denote the aggregate (or link) capacity $C_A := NC$. Note that the DE model takes C as a parameter, while NS3 sets the link bandwidth using C_A and divides it by N to set the sending rate of each flow. Note that in all calculations above, C may have to be scaled by a maximum segment size (MSS) and other constants: in all our simulations, we use the unit of segments/sec for C.

A. TCP CUBIC

Figure 4 compares the average *cwnd* generated by the NHPL simulations against the average value of *cwnd* generated by the DEs. The fixed-point value of *cwnd*, $\hat{W}$, is also shown (albeit sometimes entirely hidden by the DE curve because of fast convergence). All flows in this figure have a per-flow capacity of 1 Gbps, while the round-trip time is varied (these combinations of C and τ are sufficient to generate a diverse set of behaviors). All flows have $b = 0.2$ and $c = 0.4$ (the default values used in Linux implementations of CUBIC).

Figure 4a shows a single stable flow with $\tau = 1$ ms. The transient response of both simulations is clearly visible, and we observe that they reach steady-state within a similar period. Not shown in this panel is the value of $\hat{s} \approx 4$ seconds. By observing the time between losses in the NHPL simulation, we see that there is a close agreement. Figure 4b shows the same experiment, but with 20 flows. As expected, the average value of *cwnd* from the NHPL simulation approaches $\hat{W}$ as the number of flows increases. Figures 4c and 4d show one and 20 flows, respectively, for $\tau = 10$ ms. The initial conditions (values of $s(0)$ and $W_{\max}(0)$) are deliberately far enough from the fixed point to demonstrate a more dramatic transient response. Figure 5 shows two examples of 100 ms flows: in (a), there is a single flow that is stable, while the initial conditions in (b) cause instability for one flow in both the DEs and NHPL simulation.

Figure 6 illustrates the transient and steady-state responses of a flow with $C = 100$ Mbps and $\tau = 10$ ms, as well as

$||\mathbf{x}||_2$ as it compares to the convergence bound (18). Observe that $||\mathbf{x}||_2$ always lies below the bound and approaches zero as the flow reaches steady state. The bound appears flat in this example because for this system, $V(t_0)$ dominates in the denominator. We observe this phenomenon for many systems; this implies that the initial conditions are crucial for a flow's stability.

B. H-TCP

We present experiments with varying BDPs, ranging from low – wherein H-TCP operates exclusively in the linear regime, to high – where H-TCP's average congestion epoch is larger than Δ^L seconds. Since NS3 does not use RTT scaling, we set $\gamma = 1$ in all DE simulations. All NS3 curves are averages of five runs. Figure 7 presents comparisons of the MWLI DE model for H-TCP, using Eqs. (25) and (26) as loss probability function and queue model, respectively, against NS3 with PI as the AQM scheme. The experiment in Figure 7a uses a link capacity C_A of 500 Mbps with 15 flows in NS3. This corresponds to $C = 33$ Mbps for the DE model. Packet size is set to 1000 bytes and MSS is 958 bytes. For this experiment, $\tau = 10$ ms, which causes the BDP to be low enough that on average, H-TCP operates in the low-speed regime. This is evidenced by the average time between losses, $\overline{s}$, defined as follows:

$$\overline{s} \coloneqq \frac{1}{R}\sum_{r=1}^{R}\frac{1}{N}\sum_{f=1}^{N}\frac{1}{L_{f,r}}\sum_{i=1}^{L_f} s_i^{f,r},$$

where R is the number of NS3 experiment runs, $L_{f,r}$ is the number of losses suffered by flow f during the duration of the experiment in the rth run, and $s_i^{f,r}$ is the time between the $(i-1)$th and ith loss of flow f in the rth run (for $i = 1$, we compute the time until the first loss). For Figure 7a, $\overline{s} = 0.29$ s, which is below the low-speed regime threshold $\Delta^L = 1$ s. The DE model outputs a value of 0.27 s for $\hat{s}$.

For the experiment in Figure 7b, we increase the link capacity to one Gbps and the number of flows to 25. Here, NS3 yields $\overline{s} = 0.25$ s and the DE model yields $\hat{s} = 0.27$ s. For the experiment in Figure 7c, we use $C_A = 1$ Gbps and a propagation delay of 20 ms. The number of flows is set to 45 in NS3. Here, $\overline{s} = 0.38$ s and $\hat{s} = 0.44$ s. In Figure 7d, $C_A = 2$ Gbps and $\tau = 40$ ms, which allows H-TCP to be able to spend time in the high-speed regime, with $\overline{s} = 2.19$ s and $\hat{s} = 2.2$ s. Finally, Figure 8 presents an experiment with a relatively larger BDP compared to those of Figure 7: here, $C_A = 4$ Gbps and $\tau = 30$ ms. There are 25 NS3 flows and as with experiments in Figure 7, the average is plotted over five runs. In this plot, $\overline{s} = 1.7$ s and $\hat{s} = 1.8$ s. Note that for this larger BDP, the NS3 curve is slower to converge to the DE curve, compared to some of the experiments in Figure 7. Further, note that as the BDP increases (*e.g.*, Figure 7d and Figure 8), the DE model predicts a lack of convergence (which may be interpreted as a type of instability in the context of TCP) for H-TCP's *cwnd*. Similarly, the average NS3 flow experiences greater variation compared to *cwnd*s in lower-BDP settings. We observe this in general for larger BDPs, but predictably, the *cwnd* can be stabilized by introducing more flows, albeit at the cost of decreasing each flow's throughput and often forcing the protocol to operate in the low-speed regime. These observations are consistent with those of the numerical analysis discussion in Section V-B. Note that in all our experiments, the average *cwnd*s match closely between NS3 and the DE model, and similarly, there is close agreement between $\overline{s}$ and $\hat{s}$. Hence, Figures 7 and 8 demonstrate the MWLI model's ability to effectively predict H-TCP's behavior.

VIII. Conclusion

The main contribution of this work is a novel and versatile fluid model, which we call the MWLI model, for *cwnd*- and rate-based data transport algorithms. The model is structured so that the differential equations do not depend on the specific window or rate function of a congestion controller. As a result, this framework offers opportunities to model and analyze the stability of a diverse set of controllers whose window or rate functions may not be linear and whose increase and decrease rules may not be given in explicit form. We applied this model to three different algorithms: TCP Reno, CUBIC and H-TCP. For the former, we proved in prior work that the new model is equivalent to a well-established model for Reno. For CUBIC, the new model succeeds where traditional methods of modeling *cwnd* are ineffective. In prior work, we analyzed the fluid model for CUBIC and discovered that for a given probability of loss model, its window is locally uniformly asymptotically stable. We also derived a convergence bound on the solution of the system as a function of the system parameters. Further, we developed an event-based simulation framework to validate the model and related theoretical results for CUBIC.

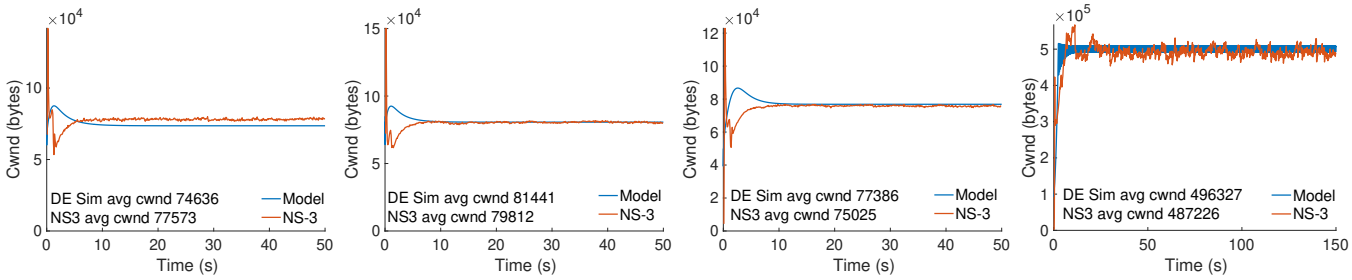
For H-TCP, we performed a linear stability analysis of the congestion controller under certain assumptions. We find that H-TCP is unstable in most cases under a particular loss model, but that it can be stabilized under a more realistic loss model, *e.g.*, such as one whose operation is closer to that of the PI AQM scheme. We validated H-TCP's model using NS3 and find that the MWLI model is able to predict average *cwnd* and average time between losses well.

Appendix A

We introduce a method of simulating the evolution of a congestion window given $W(t)$ – *cwnd* as a function of time, and $\lambda(t)$ – loss rate as a function of time. We first describe the procedure for generating loss events given arbitrary $W(t)$ and $\lambda(t)$. We then consider a specific loss model and discuss the workarounds necessary when dealing with capacity constraints and time delays. The final result is an algorithm whose pseudocode we present in detail. Finally, we illustrate the operation of the algorithm using an example *cwnd* trajectory.

A. Generating Loss Events

We would like to generate inter-loss times given a loss rate function $\lambda(t)$. In order to do so, we apply the Inverse Transform Method on the Poisson distribution, described in the following proposition.



(a) $C_A = 500$ Mbps, $\tau = 10$ ms, $q_{ref} = 500$, $q_{\max} = 1000$, $N = 15$, MSS $= 958$

(b) $C_A = 1$ Gbps, $\tau = 10$ ms, $q_{ref} = 800$, $q_{\max} = 3500$, $N = 25$, MSS $= 958$

(c) $C_A = 1$ Gbps, $\tau = 20$ ms, $q_{ref} = 1000$, $q_{\max} = 4000$, $N = 45$, MSS=958

(d) $C_A = 2$ Gbps, $\tau = 40$ ms, $q_{ref} = 800$, $q_{\max} = 2500$, $N = 20$, MSS $= 958$

Fig. 7: Validation of H-TCP's DE model using NS3. Red curves represent average *cwnd* over all NS3 flows. NS3 uses PI as the AQM scheme and DE models use Eqs. (25) and (26). Aggregate capacity is given by $C_A := NC$. q_{ref} and $q_{\max}$ have units of packets, while MSS is in bytes. Each red curve is an average of five runs of the experiment.

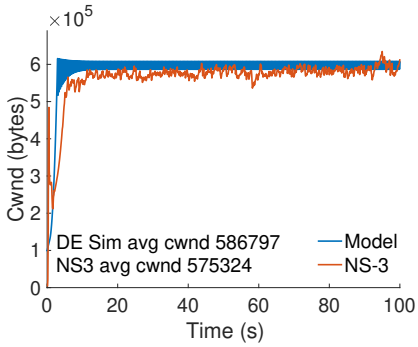


Fig. 8: Validation of H-TCP's DE model using NS3. Aggregate capacity $C_A = 4$ Gbps, $\tau = 30$ ms, $q_{ref} = 100$ packets, $q_{\max} = 1000$ packets, $N = 25$ flows, and MSS $= 1158$ bytes. The red curve is an average of five runs of NS3.

Proposition A.1. *Suppose a loss event occurs at time t_0. The time to the next loss is given by T where*

$$\int_{t_0}^{t_0+T} \lambda(t)dt = -\ln u,$$

where u is randomly generated from the uniform distribution $U(0,1)$.

Proof. Note that $\lambda(t)$ denotes a Non-Homogeneous Poisson Process, where the number of events between s and t, $N_s(t)$ has a Poisson distribution with parameter $m_s(t) = \int_s^t \lambda(\tau)d\tau$,

$$P(N_s(t) = k) = \frac{m_s(t)^k}{k!} e^{-m_s(t)}.$$

We can then write the CDF of the time from t_0 to the next loss as

$$F_{X_{t_0}}(T) = 1 - P(N_{t_0}(T) = 0) = P(N_{t_0}(T) > 0)$$
$$= 1 - \exp\Big(-\int_{t_0}^{t_0+T} \lambda(t)dt\Big).$$

Note that a CDF can be seen as a random variable with uniform distribution $U(0,1)$, and can be sampled by generating uniform random numbers (this is known as Inverse Transform Sampling). Therefore, inter-loss time samples can be generated as $T = F^{-1}_{X_{t_0}}(u)$. From the above equation we obtain

$$\int_{t_0}^{t_0+T} \lambda(t)dt = -\ln(1-u) \equiv -\ln u,$$

where the last equivalence follows from the fact that if u is uniformly distributed between 0 and 1, so is $1-u$. □

B. Delays and Capacity Constraints

In TCP (and most other data transport protocols), the loss rate is a function of the sending rate $W(t)/\tau$ and of a probability of loss model $p(t)$:

$$\lambda(t) = \frac{W(t)p(t)}{\tau}. \tag{27}$$

Therefore, in order to obtain a sample of the time until next loss, the following equation can be solved for T:

$$\frac{1}{\tau}\int_{t_0}^{t_0+T} W(t)p(t)dt = -\ln(u). \tag{28}$$

Note that $W(t)$ and $p(t)$ are viewed from the perspective of the congestion point (*e.g.* a router) where the loss is being generated. Therefore, whenever a loss occurs, the subsequent reduction in the window size (multiplicative decrease) is not reflected in $W(t)$ until after a delay of approximately τ seconds. This is illustrated in Figure 9, which shows an example trajectory of the *cwnd* function. Each time a loss i occurs at time l_i at a congestion point, a corresponding loss indication is reflected in $W(t)$ at time $T_i = l_i + \tau$. The caveat of using (28) to compute T is that $W(t)$ may have changed sometime in the time interval $[t_0, t_0+T]$ (which can happen if a loss indication is scheduled in this interval; we call this a *pending loss indication* (PLI)). In such a case, the solution is to project the current $W(t)$ until the next loss indication, update $W(t)$ to a new function, and use this new function to generate a new loss event. Once a new loss event is generated, the process may need to repeat until we either produce a loss event that takes place before the next PLI or until we run out of PLIs.

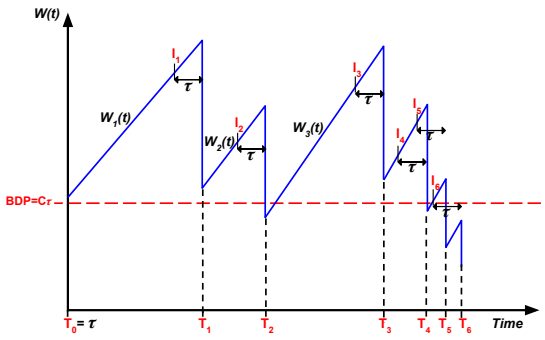


Fig. 9: Example trajectory of Reno's congestion window. l_i is the time when loss occurs at the congestion point (*e.g.* router). T_i is the time of the ith loss indication.

Another complication may arise with certain probability of loss models. For example, in this work we consider the following model:

$$p(t) = \left(1 - \frac{C\tau}{W(t)}\right)^+.$$

As a consequence, $\lambda(t) = 0$ whenever $W(t) < C\tau$. This is depicted in Figure 9, where losses only occur when $W(t) \geq C\tau$. In order to obtain an analytical solution for T during the ith loss event, we can first compute T_{BDP}, the time at which $W(t)$ reaches $C\tau$, or the bandwidth-delay product (BDP). Then, let $t_0 = \max(T_{BDP}, l_i) - T_{i-1}$, where T_{i-1} is the time of the most recent loss indication and l_i is the time of the most recent loss event at the congestion point.

Another feature of the simulation framework is the ability to generate multiple parallel flows. This feature is especially important for validating models that use a system of differential equations to characterize the behavior of congestion control algorithms. The output of such models (*e.g. cwnd*) usually describes the behavior of the average flow in a large population of flows. Indeed, in Section VII, we note that the average *cwnd* size from simulation results matches closer to the steady-state value of the DE models as we increase the number of flows in the simulation.

When multiple flows are involved, T_{BDP} is the time at which the sum of their congestion windows reaches the BDP, and l_i is the time at which the most recent loss (across all flows) occurred. We must compute t_0 for each flow, which is given by

$$t_{0,f} = \max(T_{BDP}, l_i) - T_{i-1,f},$$

where $T_{i-1,f}$ is the most recent loss indication of flow f. T is then computed using the following equation:

$$\frac{1}{\tau}\sum_{f=1}^{N}\int_{t_{0,f}}^{t_{0,f}+T} W_f(t)p_f(t)dt = -\ln(u). \qquad (29)$$

Any time a new loss event is generated, we must also choose a flow that will suffer the loss. The flow is picked based on its congestion window size at the time the loss is scheduled to occur (flows with larger windows are more susceptible to suffer a loss).

C. Pseudocode

Loss generation can be described by the pseudocode in `GeneratePoiLoss`. This function is called from the main procedure each time a loss is occurring at the congestion point in a given interval. (So, for the example in Figure 9, `GeneratePoiLoss` would be called in the intervals containing the events l_i, $i \in \{1, \ldots, 6\}$.) The arguments of the function are as follows: $pendingLITs$ is a two-dimensional matrix whose first row is a list of pending loss indication times, and whose second row contains the corresponding flows that will suffer the losses. $LLIs$ is an array that keeps record of the last loss indication times of all flows. $GLLI$ is the most recent loss indication. T_l is the time of the most recent loss event. W_{loss} is an array containing the *cwnd* sizes of all flows immediately before their most recent loss events. $p(t)$ is a probability of loss function and τ is the round-trip time. For the example in Figure 9, where there is only one flow, the procedure outlined in the pseudocode would do the following:

function GENERATEPOILOSS($pendingLITs$, $LLIs$, $GLLI$, T_l, W_{loss}, $p(t)$, τ)
  ▷ LLT: last loss time at congestion point
  ▷ $pendingLITs$: a list of pending loss indication times and corresponding flows
  Initialization:
  $GNPLI \leftarrow pendingLITs.nextLossTime$ ▷ next (global) pending loss indication time
  $LF \leftarrow pendingLITs.nextFlow$ ▷ the corresponding flow of the next loss event
  $T_{BDP} \leftarrow$ time when sum of *cwnd*'s reaches BDP
  $t_0 \leftarrow \max(T_{BDP}, T_l)$

  $lossTime \leftarrow$ COMPUTET($LLIs, GLLI, W_{loss}, t_0, \tau, p(t)$)

  while $lossTime \geq GNPLI$ **do**
    ▷ next loss occurs after $GNPLI$, so:
    ▷ (1) determine duration of current congestion epoch for flow LF:
    $I \leftarrow GNPLI - LLIs(f)$
    ▷ (2) the window function is changed at $GNPLI$, and we are looking at a new congestion epoch, so update relevant variables
    $W_{loss} \leftarrow W_{LF}(I)$ ▷ get the W_{loss} value of the next congestion epoch for flow LF
    $GLLI \leftarrow GNPLI$
    $LLIs(LF) \leftarrow GLLI$
    if $NPLI.isEmpty$ **then**
      $NPLI \leftarrow \infty$
    else
      $GNPLI \leftarrow pendingLITs.nextLossTime$
      $LF \leftarrow pendingLITs.nextFlow$
    end if
    ▷ (3) generate a new loss event at congestion point
    Recompute T_{BDP}
    $t_0 \leftarrow \max(T_{BDP}, GLLI)$
    $lossTime \leftarrow$ COMPUTET($LLIs, GLLI, W_{loss}, t_0, \tau, p(t)$)
  end while

  ▷ schedule the next loss indication event
  $pendingLITs.add(lossTime + \tau)$
  return $(lossTime, pendingLITs)$
end function

function COMPUTET($LLIs$, $GLLI$, W_{loss}, t_0, τ, $p(t)$)
  $u \leftarrow rand()$ ▷ generate a number from uniform distr.
  construct $W_f(t)$, $\forall f \in \{1, \ldots, N\}$ using W_{loss}
  $t_{0,f} \leftarrow t_0 - LLIs(f)$, $\forall f \in \{1, \ldots, N\}$
  ▷ to generate the next loss interval:
  Use Equation (29) to compute T, keep only real, positive roots
  $lossTime \leftarrow GLLI + t_0 + T$
end function

1) At time $t = 0$, a loss occurred at the congestion point (not shown in the figure), so a pending loss indication was scheduled for $T_0 = \tau$.
2) Also at the time of the loss (at $t = 0$), a new loss time was generated using `GeneratePoiLoss`. This loss time is l_1. Since l_1 occurs after the next pending loss indication (which is at T_0), the `while` loop in `GeneratePoiLoss` is triggered. We integrate the *cwnd* function from $t = 0$ to $T_0 = \tau$, compute a new W_{loss} (which is the size of the window right before T_0), and feed these values as parameters to `computeT`. The latter function computes the next loss arrival time; this is a new value of l_1. We compare this new l_1 to the next pending loss indication time (which in this case is ∞ since no other pending loss indications have been scheduled after T_0). Since $l_1 < \infty$, we exit the loop and have a new loss time of l_1 and pending loss indication $T_1 = l_1 + \tau$.
3) The main procedure iterates until it reaches the interval containing l_1, at which point `GeneratePoiLoss` is called. The latter function generates l_2, and since l_2 occurs after the next pending loss indication time (T_1), we re-generate l_2 using the same procedure as for l_1.
4) The main procedure continues until it reaches the interval containing T_1, at which point the loss indication is processed (the *cwnd* is halved and there is a new W_{loss}).
5) Loss events l_2 and l_3 and pending loss indications T_2 and T_3 are processed similarly.
6) At loss event l_4, a new loss time l_5 is generated. Since it appears before T_4, we simply schedule a pending loss at T_5 (no need to go through the `while` loop in `GeneratePoiLoss` as we did for the other losses).
7) At loss event l_5, l_6 is generated, but it occurs after the pending loss indication at T_4, which has not been processed yet. Hence, the `while` loop is triggered.

APPENDIX B

First, let us show that for H-TCP, $\Delta_0/Q = \mathcal{O}((C^2\tau^4)^{1/3})$. This is because

$$\frac{\Delta_0}{Q} = \frac{2^{1/3}\Delta_0}{(\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3}} = \frac{(\Delta_1 - \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3}}{2^{1/3}} < \Delta_1^{1/3} = \mathcal{O}(C^{2/3}\tau^{4/3}).$$

Next,

$$\frac{\Delta_0}{Q} + Q = \frac{(\Delta_1 - \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3} + (\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3}}{2^{1/3}} < \frac{2(\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3}}{2^{1/3}} = \mathcal{O}((C^2\tau^4)^{1/3}).$$

We now examine the limiting behavior of q'/S. Letting $c_0 = \Gamma + (57^3 + 228(105))/8$,

$$\frac{q'}{S} = \frac{6C\tau^2 - c_0}{\frac{1}{2}\sqrt{\frac{2}{3}p' + \frac{1}{3}\left(Q + \frac{\Delta_0}{Q}\right)}} < \frac{12\sqrt{3}C\tau^2}{\left(Q + \frac{\Delta_0}{Q}\right)^{1/2}} = \frac{12\sqrt{3}\sqrt[6]{2}C\tau^2}{\left((\Delta_1 - \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3} + (\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3}\right)^{1/2}} < \frac{12\sqrt{3}\sqrt[6]{2}C\tau^2}{\Delta_1^{1/6}} < \frac{25C\tau^2}{(C^2\tau^4)^{1/6}},$$

where the last inequality holds for large C since the coefficient of $C^2\tau^4$ in Δ_1 is 972. It is now easy to see that $q'/S = \mathcal{O}((C^2\tau^4)^{1/3})$. Finally, it remains to show that (24) is real and positive for large C and $C \gg \tau$. Clearly, under these assumptions, $S > 57/4$. It then remains to show that the quantity inside the square root is positive, or

$$2p' + q'/S > 4S^2 = \frac{2}{3}p' + \frac{1}{3}\left(Q + \frac{\Delta_0}{Q}\right).$$

Defining $y \coloneqq Q + \Delta_0/Q$, multiplying both sides by S and simplifying yields

$$6\sqrt{3}q' > (y - 4p')(2p' + y)^{1/2}.$$

Both sides of the inequality are positive for large C, so we may square them to obtain, after simplifying,

$$108(q')^2 > y^3 - 6p'y^2 + 32(p')^3.$$

For large C, it is easy to see that $y^3 - 6p'y^2 + 32(p')^3 < y^3$, so it is sufficient to prove that $108(q')^2 > y^3$, or

$$216(q')^2 > \left[(\Delta_1 - \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3} + (\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3})^{1/3}\right]^3.$$

We first focus on the right-hand side of this inequality. It is easy to see that $\Delta_1 + \sqrt{\Delta_1^2 - 4\Delta_0^3} < 2\Delta_1$. For large C and $C \gg \tau$, it is also true that $\Delta_1 - \sqrt{\Delta_1^2 - 4\Delta_0^3} < \Delta_1/8$. To see that this is true, note that this claim is equivalent to that of

$$7\Delta_1/8 < \sqrt{\Delta_1^2 - 4\Delta_0^3}, \quad \text{or} \quad 15\Delta_1^2/64 > 4\Delta_0^3,$$

and recall that Δ_1 grows with $(C\tau^2)^2$ while Δ_0 grows with $C\tau^2$. Hence, it is sufficient for us to prove that

$$216(q')^2 > \left[(\Delta_1/8)^{1/3} + (2\Delta_1)^{1/3}\right]^3 = \Delta_1(2^{1/3} + 1/2)^3.$$

Noting that $(2^{1/3} + 1/2)^3 < 6$, it is sufficient to show that $36(q')^2 > \Delta_1$, or, after expanding both sides,

$$1296(C\tau^2)^2 - 432c_0C\tau^2 + 36c_0^2 > 972(C\tau^2)^2 - 348574.3C\tau^2 - 571698\tau^2 + 2070623.3,$$

which easily holds for large C and $C \gg \tau$.

ACKNOWLEDGMENT

This work was supported by the US Department of Energy under Contract DE-AC02-06CH11357 and by the National Science Foundation under Grant No. CNS-1413998.

REFERENCES

[1] T. Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," *ACM SIGCOMM computer communication Review*, 2003.
[2] S. Ha, I. Rhee, and L. Xu, "CUBIC: a New TCP-Friendly High-Speed TCP Variant," *ACM SIGOPS Operating Systems Review*, 2008.
[3] D. Leith and R. Shorten, "H-TCP: TCP for high-speed and long-distance networks," in *Proceedings of PFLDnet*, 2004.
[4] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *Queue*, 2016.

[5] ESnet, "ESnet Fasterdata Knowledge Base," http://fasterdata.es.net/, Accessed: 2018-10-31.
[6] N. S. Rao, N. Imam, J. Hanley, and S. Oral, "Wide-Area Lustre File System Using LNet Routers," in *2018 Annual IEEE International Systems Conference (SysCon)*. IEEE, 2018, pp. 1–6.
[7] Y. Gu and R. L. Grossman, "UDT: UDP-based Data Transfer for High-speed Wide Area Networks," *Computer Networks*, 2007.
[8] R. Pan, B. Prabhakar, and A. Laxmikantha, "QCN: Quantized Congestion Notification," *IEEE802*, 2007.
[9] G. Vardoyan, C.V. Hollot, D. Towsley, "Towards Stability Analysis of Data Transport Mechanisms: a Fluid Model and an Application," *arXiv preprint arXiv:1801.02741*, 2018.
[10] NS-3 Development Team. NS-3 Network Simulator, https://www.nsnam.org/, Accessed: 2018-01-10.
[11] G. Vardoyan, C. Hollot, and D. Towsley, "Towards Stability Analysis of Data Transport Mechanisms: a Fluid Model and an Application," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 666–674.
[12] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED," *SIGCOMM Comput. Commun. Rev.*, 2000.
[13] F. P. Kelly, A. K. Maulloo, and D. K. Tan, "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability," *Journal of the Operational Research society*, 1998.
[14] R. Srikant, *The Mathematics of Internet Congestion Control*. Springer Science & Business Media, 2012.
[15] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "A Control Theoretic Analysis of RED," in *INFOCOM 2001.*, 2001.
[16] X. Huang, L. Chuang, and R. Fengyuan, "Generalized Modeling and Stability Analysis of Highspeed TCP and Scalable TCP," *IEICE transactions on communications*, 2006.
[17] W. Bao, V. Wong, and V. Leung, "A Model for Steady State Throughput of TCP CUBIC," in *GLOBECOM 2010*.
[18] S. Poojary and V. Sharma, "An Asymptotic Approximation of TCP CUBIC," *arXiv preprint arXiv:1510.08496*, 2015.
[19] S. Ha, Y. Kim, L. Le, I. Rhee, and L. Xu, "A step toward realistic performance evaluation of high-speed tcp variants," in *Fourth International Workshop on Protocols for Fast Long-Distance Networks*, 2006.
[20] D. X. Wei, C. Jin, S. H. Low, and S. Hegde, "Fast tcp: motivation, architecture, algorithms, performance," *IEEE/ACM transactions on Networking*, vol. 14, no. 6, pp. 1246–1259, 2006.
[21] G. Wang, Y. Wu, K. Dou, Y. Ren, and J. Li, "AppTCP: The design and evaluation of application-based TCP for e-VLBI in fast long distance networks," *Future Generation Computer Systems*, vol. 39, pp. 67–74, 2014.
[22] K. Gu, J. Chen, and V. L. Kharitonov, *Stability of Time-Delay Systems*. Springer Science & Business Media, 2003.
[23] D. Leith, R. Shorten, and Y. Lee, "H-TCP: A framework for congestion control in high-speed and long-distance networks," in *PFLDnet Workshop*, 2005.
[24] D. Leith and R. Shorten, "On RTT Scaling in H-TCP," *Discussion note, Hamilton Institute*.
[25] D. Leith, "H-TCP: TCP Congestion Control for High Bandwidth-Delay Product Paths," Internet Requests for Comments, Tech. Rep., April 2008. [Online]. Available: https://tools.ietf.org/id/draft-leith-tcp-htcp-06.txt
[26] R. C. Dorf and R. H. Bishop, *Modern Control Systems*. Pearson, 2011.
[27] D. Genin and V. Marbukh, "Bursty fluid approximation of tcp for modeling internet congestion at the flow level," in *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2009, pp. 1300–1306.
[28] C. V. Hollot, V. Misra, D. Towsley, and W. Gong, "Analysis and Design of Controllers for AQM Routers Supporting TCP Flows," *IEEE Transactions on automatic control*, vol. 47, no. 6, pp. 945–959, 2002.
[29] C. V. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "On Designing Improved Controllers for AQM Routers Supporting TCP Flows," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No. 01CH37213)*, vol. 3. IEEE, 2001, pp. 1726–1734.
[30] R. Pan, P. Natarajan, F. Baker, and G. White, "Proportional Integral Controller Enhanced (PIE): A Lightweight Control Scheme to Address the Bufferbloat Problem," Tech. Rep., 2017. [Online]. Available: https://tools.ietf.org/html/rfc8033
[31] C. Hollot and Y. Chait, "Nonlinear stability analysis for a class of TCP/AQM networks," in *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*, vol. 3. IEEE, 2001, pp. 2309–2314.