Models and Methods for Analyzing Internet Congestion Control Algorithms

R. Srikant

Department of Electrical and Computer Engineering and
Coordinated Science Lab
University of Illinois at Urbana-Champaign
Urbana, IL 61801
rsrikant@uiuc.edu

1 Introduction

Congestion control in the Internet was introduced in the late 1980s by Van Jacobson [0]. Jacobson's algorithm which is implemented in the transport layer protocol called TCP (Transmission Control Protocol) has served the Internet well during a time of unprecedented growth. However, the algorithm was designed during a time when the Internet was a relatively small network compared to its size today. Therefore, there has been much interest in reexamining the role of congestion control in the Internet with the goal of enhancing TCP to make it scalable to large networks.

We begin this survey with a simple model of a particular version of Jacobson's algorithm called TCP-Reno. We show that the algorithm may not be stable when the network size becomes large, i.e., if the feedback delays are large or the capacity of the network is large. We then present Kelly's model where congestion control is viewed as a distributed control algorithm for achieving fair resource allocation in a network with many users. The remarkable feature of the algorithm is that the complex interactions between the users in the network can be captured using quantities that can be measured easily by each individual user. Next, we study the stability of this congestion control mechanism in the presence of feedback delay. We primarily concentrate on a new technique using Razumikhin's theorem that proves global stability of the congestion controllers under some assumptions on the control parameters. The interested reader may refer to the results in [0, 0, 0, 0, 0, 0] where local stability results are obtained without restrictions on the network parameters. However, the region of attraction for these local stability results is yet to be established for a general topology network. Finally, we turn our attention to connection-level models of congestion control. In traditional congestion control models, the number of users on the various routes in the network is assumed to be fixed. In the connection-level model, the number of files in the network

is time-varying and is described by a Markov chain. Between the time instants when there is a file arrival or departure, it is assumed that congestion control occurs instantaneously. Thus, the connection-level model operates at a slower time-scale compared to the time required for congestion controllers to converge. Under this assumption of time-scale separation, we show that the Kelly model of resource allocation is efficient, i.e., if the total load on each link in the network is less than its capacity, then the network is stable.

The rest of this paper is organized in follows. In Section , we present a simple model for Jacobson's algorithm that was introduced in [0] and analyze its local stability in the presence of feedback delay. We summarize the results in [0, 0] which show that TCP-Reno is not stable when the capacity per user is large or if the feedback delay is large. In Section , we introduce the Kelly resource allocation model [0] and the associated congestion control algorithms [0, 0]. A stability analysis of the congestion control algorithm in the presence of feedback delays is presented in Section . This stability result was obtained in [0] using Razumikhin theorem. A related result has also been obtained in [0] using different techniques. In Section , we present the connection-level model introduced in [0] and present a stability result using a drift analysis. This result which was first established in [0] using fluid limits and a special case of this result was presented earlier in [0]. Finally, concluding remarks are presented in Section . The survey presented here covers only a small portion of the considerable research on Internet Congestion Control that has taken place over the last few years. For a more comprehensive survey, we refer the reader to [0].

2 A Simple Model for TCP-Reno

Congestion control is implemented in the Internet using a *window flow control* algorithm. A source's *window* is the maximum number of unacknowledged packets that the source can inject into the network at any time. For example, if the window size is 1, then the source maintains a counter which has a maximum value of 1. The counter indicates the number of packets that it can send into the network. The counter's value is initially equal to the window size. When the source sends one packet into the network, the counter is reduced by 1. Thus, the counter in this example would become zero after each packet transmission and the source cannot send any more packets into the network till the counter hits 1 again. To increment the counter, the source waits for the destination to acknowledge that it has received the packet. This is accomplished by sending a small packet called the *ack* packet, from the destination back to the source. Upon receiving the ack, the counter is incremented by 1 and thus, the source can again send one more packet. We use the term *round-trip time (RTT)* to refer to the amount of time that elapses between the instant that the source transmits a packet and the instant at which it receives the acknowledgment for the packet. The RTT consists of

three components: the propagation delay of the packet through the transmission medium (which is determined by the distance between the source and destination), the queueing delay at the routers in the network and the time taken to process a packet at the routers in the network. Typically, the processing time is negligible compared to the other two components. With a window size of 1, since one packet is transmitted during every RTT, the source's data transmission rate is $1/RTT$ packets/sec.

If the window is 2, the counter's value is initially set to 2. Thus, the source can send two back-to-back packets into the network. For each transmitted packet, the counter is decremented by 1. Thus, after the first two packet transmissions, the counter is decremented to zero. When one of the packets is acknowledged and the ack reaches the source, then the source increments the counter by 1 and can send one more packet into the network. Once the new packet is transmitted, the counter is again decremented back to zero. Thus, after each ack, one packet is sent, and then, the source has to wait for the next ack before it can send another packet. If one assumes that the processing speed of the link is very fast and that the processing times at the source and destination are negligible, then the source can transmit two packets during every RTT. Thus, the source's transmission rate is $2/RTT$ packets/sec. From the above argument, it should be clear that, if the window size is W, then the transmission rate can be approximated by W/RTT packets/sec.

If the link capacity is c and the source's window size W is such that $W/RTT < c$, then the system will be stable. In other words, all transmitted packets will be eventually processed by the link and reach the intended destination. However, in a general network, the available capacity cannot be easily determined by a source. The network is shared by many sources which share the capacities at the various links in the network. Thus, each source has to adaptively estimate the value of the window size that can be supported by the network. The most widely-used algorithm for this purpose in the Internet today is called TCP-Reno.

The TCP-Reno algorithm is quite complicated and therefore, for our modelling purposes, we consider the following simplified version of the algorithm. Assume that there is a mechanism for the receiver to indicate to the source that a packet has been lost in the network. Then, the essential features of the TCP-Reno algorithm can be summarized as below:

- Upon receipt of a ack, the source increases its current window size, denoted by *cwnd*, as follows:

$$cwnd \leftarrow cwnd + 1/cwnd.$$

- Upon being informed of a loss, the source decreases its window size by a factor of two:

$$cwnd \leftarrow cwnd/2.$$

The key feature of TCP-Reno is that it increases its window size when it does not detect congestion which is indicated by the reception of an ack, and it

decreases its window size upon detecting congestion, which is indicated by the detection of a lost packet.

We now present a differential equation model that describes the TCP-Reno congestion control algorithm. Consider N TCP-Reno sources, all with the same RTT, accessing a single link. Let $W_r(t)$ denote the window size of flow r, T be its RTT, and $q(t)$ be the fraction of packets lost at the link at time t. Then, the congestion avoidance phase of TCP-Reno can be modelled as

$$\dot{W}_r = \frac{x_r(t-T)(1-q(t-T))}{W_r} - \beta x_r(t-T)q(t-T)W_r(t), \tag{1}$$

where $x_r(t) = W_r(t)/T$ is the transmission rate. The parameter β is the decrease factor and is taken to be $1/2$ although studies show that a more precise value of β when making a continuous-time approximation of TCP's behavior is closer to $2/3$. Substituting for $W_r(t)$ in terms of $x_r(t)$ gives

$$\dot{x}_r = \frac{x_r(t-T)(1-q(t))}{T^2 x_r} - \beta x_r(t-T)q(t)x_r(t). \tag{2}$$

The loss probability $q(t)$ is a function of the arrival rate at the link. Thus, let

$$q(t) = f(y(t-T)),$$

where $f(\cdot)$ is an increasing function and $y(t)$ is the total arrival rate at the link and is given by

$$y(t) = \sum_{r=1}^{N} x_r(t).$$

The equilibrium value of x_r is easily seen to be

$$\hat{x}_r = \sqrt{\frac{1-\hat{q}}{\beta\hat{q}}}\frac{1}{T}, \tag{3}$$

where $\hat{q}$ is the equilibrium loss probability. We note that throughout this paper, we use $\hat{}$ to denote equilibrium values. The functional form of $f(y)$ could be quite complicated in general. Among other things, it will depends on the assumptions on the stochastic behavior of the packet arrival process at the router. To simplify the analysis, we will assume that $f(y)$ is of the form suggested in [0]:

$$f(y) = \left(\frac{y-c}{y}\right)^+.$$

Thus, this form of $f(y)$ can be interpreted as a fluid approximation to the loss probability: it is equal to zero if the arrival rate is less than the capacity of the link and is otherwise equal to the fraction by which the arrival rate exceeds the link capacity. Recall that the RTT T consists of two components, namely the propagation delay T_p and the queueing delay at the router. Just like the

loss probability, it is difficult to precisely capture the queueing delay using a simple analytical formula. To obtain a tractable expression for the queueing delay, we recall that the TCP-Reno protocol attempts to fill up the buffer at the router and uses the resulting packet loss to obtain congestion information. Therefore, it seems reasonable to assume that the queue is full most of time. Under this assumption, our approximation to the queueing delay takes the form B/c, where B is the buffer size at the router. Thus, for all users, the RTT is given by

$$T = T_q + \frac{B}{c}.$$

To study the stability of the congestion controller given in (2), we first linearize the system around its equilibrium point. Defining $\delta x_r = x_r - \hat{x}_r$, and $\delta q = q - \hat{q}$, the linearized form of the congestion control algorithm is given by

$$\delta\dot{x}_r = -\hat{x}_r \left(\frac{1-\hat{q}}{T^2 \hat{x}_r^2} \delta x_r + \frac{1}{T^2 \hat{x}_r} \delta q + \beta \hat{q} \delta x_r + \beta \hat{x}_r \delta q \right),$$

and

$$\delta q = \frac{c}{\hat{y}^2} \sum_r \delta x_r (t - T).$$

Defining $\delta y = y - \hat{y}$, and using the equilibrium relationship (3) yields

$$\delta\dot{x}_r + \alpha_1 \delta x_r + \alpha_2 \delta x_r (t - T) = 0, \tag{4}$$

where

$$\alpha_1 = 2\beta \hat{q} \hat{x}_r, \qquad \alpha_2 = \beta \hat{x}_r.$$

From Hayes' lemma [0], the linearized delay-differential equation describing TCP-Reno's dynamics is stable if one of the following conditions is satisfied:

- $\alpha_1 \geq \alpha_2$,
- $\alpha_1 < \alpha_2$ and

$$\alpha_2 T \sqrt{1 - \frac{\alpha_1^2}{\alpha_2^2}} < \arccos\left(-\frac{\alpha_1}{\alpha_2} \right).$$

For the first condition to be satisfied, we require $\hat{q} \geq 1/2$. This is not a practical scenario since it requires at least half the packets to be dropped at the router. The second condition can be written as

$$\frac{c}{N} T < \frac{1}{\beta} \frac{(1-\hat{q}) \arccos(-2\hat{q}_r)}{\sqrt{1 - 4\hat{q}^2}}. \tag{5}$$

Note that the equilibrium relationship (3) can be rewritten as

$$\frac{(1-\hat{q})^3}{\hat{q}} = \left(\frac{c}{N} T \right)^2.$$

If we let c/N (which is simply the capacity per user) be a constant and increase the RTT, then it is clear from the previous equation that $\hat{q}$ must decrease. Thus, for large T, the right-hand side of the stability condition can be approximated by letting $\hat{q} = 0$ which gives the following condition for stability

$$\frac{c}{N}T < \frac{\pi}{2\beta}.$$

Clearly, this condition will be violated as T increases or c/N increases. From the above analysis, we can conclude that TCP-Reno is not a scalable protocol, i.e., its stability is compromised if either the RTT of the users is large or if the available capacity per user at the router is large. Related linear analysis of TCP dynamics can be found in [0, 0]. In the following sections, we will examine alternatives to TCP-Reno as a congestion control mechanism.

3 Resource Allocation and Congestion Control

Consider a network consisting of a set of links denoted by $\mathcal{L}$ and a set of users denoted by $\mathcal{R}$. User r chooses a collection of links called a route to transmit its packets from a source to a destination. We will associate a unique route with a user and therefore, an index r can be used to represent both a user and the route used by the user. The data transmission rate of user r is denoted by x_r. The capacity of each link l is denoted by c_l. Since the total arrival rate on each link should be less than or equal to the capacity of the link, we have following constraints:

$$\sum_{r:l\in r} x_r \leq c_l, \qquad \forall l, \tag{6}$$

where $l \in r$ is a notation for "link l belongs to route r." The resource allocation problem in the Internet is to find a set of rates $\{x_r\}$ which satisfy the constraints (6). However, without additional constraints, typically there would be many solutions to this problem and it would be difficult to distinguish between them. For example, consider a single link of unit capacity shared by two users. Then, any set of non-negative values x_1, x_2 satisfying $x_1 + x_2 \leq 1$ would be feasible. Even if we impose the natural condition that the link should be fully utilized, we still have an infinite number of solutions to $x_1 + x_2 = 1$. Among the many solutions, the particular solutions $x_1 = 0, x_2 = 1$ or $x_1 = 1, x_2 = 0$ are especially unappealing since they are severely unfair to one of the users. Thus, a reasonable goal to aim for in resource allocation is *fairness*.

To define a fair resource allocation problem, associate with each r a utility function $U_r(x_r)$ which indicates the utility to user r when it is allowed to transmit at rate r. Then, the fair resource allocation problem is to maximize

$$\max_{\{x_r\}} \sum_r U_r(x_r). \tag{7}$$

Thus, in this problem, a resource allocation is said to be fair if it maximizes the sum of the utilities of all the users subject to the link capacity constraints (6) and the obvious non-negativity constraints

$$x_r \geq 0, \qquad \forall r. \tag{8}$$

In the rest of this section, we will interpret congestion control as a decentralized solution to the resource allocation problem (7).

Instead of solving the resource allocation problem (7) exactly, let us consider the following objective:

$$V(\mathbf{x}) = \sum_r U_r(x_r) - \sum_l \int_0^{\sum_{s:l\in s} x_s} f_l(y)dy, \tag{9}$$

where we interpret $f_l(\cdot)$ as the *penalty function or the barrier function* [0] as the case may be, or simply as the *price* for sending traffic at rate $\sum_{s:l\in s} x_s$ on link l. We assume that the price functions $f_l(\cdot)$ are increasing functions. Under this assumption, it is easy to see that $V(\mathbf{x})$ is a strictly concave function. Now, consider the following problem:

$$\max_{\{x_r\geq 0\}} V(\mathbf{x}). \tag{10}$$

The optimal source rates satisfy

$$\frac{\partial V}{\partial x_r} = 0, \qquad \forall r.$$

This gives the following set of equations:

$$U_r'(x_r) - \sum_{l:l\in r} f_l\left(\sum_{s:l\in s} x_s\right) = 0, \qquad \forall r. \tag{11}$$

Clearly, the above set of equations is difficult to solve in a centralized manner. It requires knowledge of all the sources' utility functions, their routes and the link penalty functions f_l's. Next, we will see how congestion controllers compute the solution to (10) in a decentralized manner.

Let us suppose that each router (node) in the network has the ability to monitor the traffic on each of its links and compute y_l for each link l connected to it. Let $p_l(t)$ denote the price for using link l at time t. In other words,

$$p_l(t) = f_l(y_l(t)).$$

Define the price of a route to be the sum of the prices of the links on the route. Now, suppose that the sources and the network have a protocol that allows each source to obtain the sum of the link prices along its route. One way to convey the route prices would be to have a field in each packet's header which

can be changed by the routers to convey the price information. For instance, the source may set the price field equal to zero when it transmits a packet. Each router on the packet's route could then add the link prices to this field so that, when the packet reaches the destination, the field would contain the route price. The destination could then simply transmit this information back to the source in the acknowledgment (ack) packet.

Let q_r denote the route price on route r, i.e.,

$$q_r = \sum_{l:l\in r} p_l.$$

Using this notation, the condition (11) can be rewritten as

$$U_r'(x_r) - q_r = 0, \qquad \forall r. \tag{12}$$

Consider the following congestion control algorithm for solving the set of equations (12) [0]: for each user r,

$$\dot{x}_r = k_r(x_r)\left(U_r'(x_r) - q_r(t)\right), \tag{13}$$

where $k_r(x)$ is an appropriately chosen scaling function. Note that the right-hand side of the differential equation (13) describing the congestion control algorithm is simply the partial derivative of the objective function (9) with respect to x_r, multiplied by $k_r(x_r)$. Thus, the congestion controller is the well-known gradient algorithm that is widely used in optimization theory [0].

The following theorem shows that the congestion control algorithm is globally asymptotically stable, i.e., starting from any initial condition, in the limit as $t \to \infty$, the set of sources rates $x_r(t)$ will converge to the set of non-zero rates $\hat{x}_r$ that maximize the objective $V(\mathbf{x})$ in (9).

Theorem 1. *Assume that the utility functions $U_r(x_r)$, the price functions $f_l(y_l)$ and the scaling functions $k_r(x_r)$ are such that*

- *the problem (10) has a unique solution $\hat{x}_r$ such that $\hat{x}_r \neq 0\ \forall r$,*
- *$\hat{x}_r = 0\ \forall r$ is not an equilibrium point of (13) and*
- *$V(\mathbf{x}) \to -\infty$ as $\|\mathbf{x}\| \to \infty$.*

Then, starting from any initial condition $\{x_r(0) \geq 0\}$, the congestion control algorithm (13) will converge to the unique solution of (10)*, i.e., $\mathbf{x}(t) \to \hat{\mathbf{x}}$ as $t \to \infty$.*

Proof. Note that

$$\dot{V} = \sum_r \frac{\partial V}{\partial x_r}\dot{x}_r = \sum_r k_r(x_r)\left(U_r'(x_r) - q_r\right)^2 \geq 0.$$

Further, note that $\dot{V} > 0$ for $\mathbf{x} \neq \hat{\mathbf{x}}$ and is equal to zero for $\mathbf{x} = \hat{\mathbf{x}}$. Thus, $V(\mathbf{x})$ is a Lyapunov function for the system and by the assumptions of the theorem, the system of congestion controllers is globally, asymptotically stable and converges to the unique solution of (10).

□

In the penalty function method, the link prices p_l are computed based on price functions $f_l(x_l)$, and thus, while it solves the problem (10), it does not solve the original resource allocation problem (7) formulated in the previous section exactly. At equilibrium, the algorithm (13) can be interpreted as computing an approximation to the Lagrange multipliers of the original resource allocation problem (7) and (6). Clearly, the congestion controller will converge to the optimal solution of (7) if the equilibrium prices $\hat{p}_l$ are indeed the Lagrange multipliers. To this end, instead of computing p_l via the static relationship $p_l = f_l(y_l)$, consider the following dynamic equation to update p_l [0, 0, 0, 0]:

$$\dot{p}_l = g_l(p_l)(y_l - c_l)^+, \tag{14}$$

where it is assumed that $h(p_l) > 0$. The congestion controller (13) can be interpreted as computing the primal variables which are the source rates and the dynamic price update (14) can be interpreted as computing the dual variables or the Lagrange multipliers. Therefore, (13) and (14) together is called the primal-dual algorithm. The widely-studied dual algorithm can be viewed as a special case of the above algorithm where $k_r(x_r) \to \infty$ [0, 0]. The exact penalty function approach (also known as the adaptive virtual queue or AVQ algorithm [0, 0]) cannot be directly interpreted in terms of the primal-dual algorithm presented here, but it is an alternative method to compute the Lagrange multipliers precisely.

Define

$$H_r(x_r) := \int_{\hat{x}_r}^{x_r} \frac{u_r - \hat{x}_r}{k_r(u_r)}\, du\,,$$

$$J_l(p_l) := \int_{\hat{p}_l}^{p_L} \frac{v_l - \hat{p}_l}{g_l(v_l)}\, dv\,,$$

and let

$$V(x,p) = \sum_r H_r(x_r) + \sum_l J_l(p_l)\,.$$

Then, it can be verified that $V(x,p)$ is a Lyapunov function for the primal-dual algorithm defined by (13)-(14) and that the system is globally, asymptotically stable [0].

4 Stability in the Presence of Delays

In this section, we study the global stability of the primal congestion controller introduced in the previous section in the presence of delays. We first introduce some notation to describe the network with delays. Let $d_f(i,l)$ be the forward delay from source i to link l, and $d_b(i,l)$ be the backward delay from link l to source i. Denote by $T_i = d_f(i,l) + d_b(i,l)$ the round trip time (RTT) for source i. We assume that the RTT is a constant and not time-varying. This is a reasonable assumption if the price functions $f_l(y_l)$ are designed such that

the users get early feedback about congestion so that they react before queue buildup occurs at the routers. In that case, the RTT is simply equal to the propagation delay which is a constant.

We consider the following congestion control algorithm:

$$\dot{x}_i(t) = k_i x_i\left(t - T_i\right)\left(\frac{a}{x_i^n(t)} - b x_i^m(t) q_i(t)\right), \tag{15}$$

where

$$\begin{aligned} q_i(t) &= \sum_{l \in i} p_l(t - d_b(i,l)), \\ p_l(t) &= f_l(y_l(t)), \\ f_l(y_l(t)) &= \left(\frac{y_l(t)}{c_l}\right)^h \end{aligned} \tag{16}$$

and

$$y_l(t) = \sum_{k:l \in k} x_k(t - d_f(i,l)).$$

Here a, b, and h are positive real numbers, and m, n are real numbers that satisfy $m+n > 0$. In the above set of equations, x_i is the rate at which source i transmits data, y_l is the arrival rate at link l, p_l is price of link l, q_i is the price of source i's route and $f_l(y)$ is the function of the link arrival rate which is used to compute p_l. The price of a route is simply the sum of the prices of the links along its path. Also define the quantity $d := \max_j T_j$ which will be useful later. Note that when $d = 0$, the algorithm (15) is a special case of (13). Specifically, here we have chosen

$$U_i'(x_i) = \frac{a}{b x_i^{m+n}}$$

and the scaling function to be $k_i x_i^{m+1}$ for some constant $k_i > 0$.

It has been shown in [0] that $\mathbf{x}(t) > 0$ for all t whenever the initial conditions are non-zero. So we can define the functions

$$W_j(t) = \frac{1}{2}\left(\log x_j(t) - \log \hat{x}_j\right)^2 \tag{17}$$

which are well-defined provided that the network model has a nonzero initial condition. Also define the function

$$W(t) = \max_j W_j(t). \tag{18}$$

In what follows, we will show that there exists an $\alpha > 1$ such that $W(t)$ decreases whenever

$$\alpha^2 W(t) > \max_{t-d \le r \le t} W(r). \tag{19}$$

It then follows from Razumikhin's theorem that the set of delay-differential equations describing the primal controller is globally asymptotically stable. The next lemma shows that, if the Lyapunov function W defined by (17-18) satisfies the condition in (19) at some time instant t, then this naturally imposes upper and lower bounds on the functions x_j over the interval $\in [t-d, t]$.

Lemma 1. *Suppose the network model has a nonzero initial condition, that at time t the Lyapunov function W satisfies the condition in (19), and that index i is such that that*

$$W(t) = W_i(t).$$

Then

$$B_i^{-\beta}\hat{x}_j < x_j(r) < B_i^{\beta}\hat{x}_j, \tag{20}$$

for all $\tau \in [t-d, t]$, *where* $B_i = \frac{x_i(t)}{\hat{x}_i}$ *and* $\beta = \mathrm{sgn}(\log B_i)\alpha$.

Proof. First note that since $W_i(t) \neq 0$, we have $B_i \neq 1$. By the definition on $W_i(t)$ we have for each index j and $\tau \in [t-d, t]$ that

$$\alpha^2 \left(\log B_i\right)^2 = \alpha^2 W(t) > W_j(t) = \left(\log \frac{x_j(t)}{\hat{x}_j}\right)^2.$$

Note that above inequality can be rewritten as

$$(\log B_i^{\alpha})^2 > \left(\log \frac{x_j(t)}{\hat{x}_j}\right)^2$$

which implies that

$$-\mathrm{sgn}(\log B_i) \log B_i^{\alpha} < \log \frac{x_j(t)}{\hat{x}_j} < \mathrm{sgn}(\log B_i) \log B_i^{\alpha}.$$

From this, we get the inequalities

$$\hat{x}_j B_i^{-\beta} < x_j(r) < \hat{x}_j B_i^{\beta}.$$

□

The next lemma then shows that the route prices are also upper and lower bounded as a consequence of the previous lemma.

Lemma 2. *Under the conditions of Lemma , the price of each route i is bounded as given by the following expression:*

$$\frac{a}{b} B_i^{-h\beta} \hat{x}_i^{-m-n} < q_i(t) < \frac{a}{b} B_i^{h\beta} \hat{x}_i^{-m-n}.$$

Proof. The bounds on the source rates in the previous lemma immediately give a bound on the link arrival rates y_l, which in turn give bounds on the link prices p_l. It is then easy to see from the definition of the route price $q(t)$ that

$$\hat{q}_i B_i^{-\beta h} < q_i(t) < \hat{q}_i B_i^{\beta h}.$$

Furthermore since $\hat{q}_i = \frac{a}{b}\hat{x}_i^{-m-n}$, we can conclude that

$$\frac{a}{b} B_i^{-h\beta} \hat{x}_i^{-m-n} < q_i(t) < \frac{a}{b} B_i^{h\beta} \hat{x}_i^{-m-n}.$$

Corollary 1 *If $m+n > h$, the supposition of Lemma holds and $\alpha = (m+n+h)/2h$, then $\dot{W}_i(t) < 0$.*

Proof. The derivative $\dot{W}_i(t)$ is given by

$$\dot{W}_i(t) = \frac{\dot{x}_i(t)}{x_i(t)} \log\left(\frac{x_i(t)}{\hat{x}_i}\right) = \frac{\dot{x}_i(t)}{x_i(t)} \log B_i,$$

and thus to prove the result we need to show that $\dot{x}_i(t) \log B_i < 0$. From (15) we see that

$$\dot{x}_i(t) = k_i \frac{x_i(t-T_i)}{x_i^n(t)} \left(a - b q_i(t) x^{m+n}(t)\right).$$

Since $x_i(t-T_i)$ and $x_i(t)$ are both positive it is sufficient to show that

$$\left(a - b q_i(t) x^{m+n}(t)\right) \log B_i < 0. \tag{21}$$

We can show this using Lemma by considering the following two cases. Suppose first that $B_i > 1$, then we have to show that

$$q_i(t) x_i^{m+n}(t) > \frac{a}{b}.$$

From the lower bound in Lemma , we have

$$q_i(t) x_i^{m+n}(t) > \frac{a}{b} B_i^{-h\alpha} \hat{x}_i^{-m-n} x_i^{m+n}(t) = \frac{a}{b} B_i^{-h\alpha+m+n}$$

which is greater than a/b since $\alpha h < m+n$. Next, suppose that $B_i < 1$. Then, we have to show that

$$q_i(t) x_i^{m+n}(t) < \frac{a}{b}.$$

From the upper bound in Lemma ,

$$q_i(t) x_i^{m+n}(t) < \frac{a}{b} B_i^{-h\alpha} \hat{x}_i^{-m-n} x_i^{m+n}(t) = \frac{a}{b} B_i^{-h\alpha+m+n}$$

which is now less than a/b since $\alpha h < m+n$.

Now we state our result on the stability of congestion controllers with delay.

Theorem 1 *If $m + n > h > 0$, then the network model in (15) is globally asymptotically stable.*

Proof. With $W(t)$ and $W_j(t)$ as defined above, it is easy to show that, for every $t \geq 0$,

$$\text{limsup}_{a \to 0^+} \frac{1}{a} \{W(t+a) - W(t)\} < 0 \tag{22}$$

whenever $\alpha^2 W(t) > \max_{t-d \leq r \leq t} W(r)$. Thus, the result follows from Razumikhin's theorem [0]. □

5 Stability with File Arrivals and Departures

In this section, we study simple models of the dynamics of a network at the connection level, i.e., at the level of file arrivals and departures. In all the previous sections, we assumed that the number of controlled flows in the network is a constant and studied the congestion control and resource allocation problem for a fixed number of flows. In reality, connections or files or flows arrive and depart from the network. We assume that the amount of time that it takes for the congestion control algorithms to drive the source rates close to their equilibrium is much smaller than the inter-arrival and inter-departure times of files. In fact, we assume that compared to the time-scale of connection dynamics, the congestion control algorithm operates instantaneously, providing a resource allocation dictated by the utility functions of the users of the network.

Consider a network in which files are arriving to route r at rate λ_r files-per-second according to a Poisson process. The file sizes are independent and exponentially distributed, with the mean file size being $1/\mu_r$ on route r. Let $n_r(t)$ be the number of files on route r at time t. It is assumed that each file on route r is allocated $\hat{x}_r(t)$ bits-per-second at time t, where the rates $\{\hat{x}_r\}$ are chosen as the optimal solution to the following resource allocation problem:

$$\max_{\{x_r\}} \sum_r w_r n_r \frac{x_r^{1-\alpha}}{1-\alpha} \tag{23}$$

subject to

$$\sum_{r:l \in r} n_r x_r \leq c_l, \qquad \forall l,$$

$$x_r \geq 0, \qquad \forall r.$$

In this optimization problem, it is assumed that all files using the same route have the same utility function $w_r x_r^{1-\alpha}/(1-\alpha)$ for some $\alpha > 0$, $\alpha \neq 1$. When $\alpha = 1$, we assume that the utility function is $U_r(x_r) = w_r \log x_r$. This connection-level model assumes that, compared to the inter-arrival and departure times of files, congestion control operates at a very fast time scale so that,

at the connection time-scale, it appears as though the congestion controller converges instantaneously to solve the resource allocation problem instantaneously. Thus, a time-scale separation is assumed between congestion control (the fast time-scale) and file arrivals and departures (the slow time-scale).

Note that $\hat{x}_r$ is not uniquely determined when $n_r = 0$. For this reason, it is more convenient to describe the optimization problem in terms of new variables $\{\hat{\chi}_r\}$, where $\hat{\chi}_r$ is the total rate allocated to users on route r, i.e., $\hat{\chi}_r = n_r x_r$. These rates are obtained as solutions to the following optimization problem:

$$\max_{\{\chi_r\}} \sum_r w_r n_r^\alpha \frac{\chi_r^{1-\alpha}}{1-\alpha} \tag{24}$$

subject to

$$\sum_{r:l\in r} \chi_r \le c_l, \qquad \forall l,$$
$$\chi_r \ge 0, \qquad \forall r.$$

Again $\hat{\chi}_r$ is not uniquely determined when $n_r = 0$. However, $\hat{\chi}_r$ is always less than or equal to $\zeta_r := \max_{l:l\in r} c_l$.

Consider the load on any link l due to the file arrival process. If a route r uses link l, then on average, it requires λ_r/μ_r bits-per-second from link l. Thus, it is rather obvious that the network can support the traffic load only if

$$\sum_{r:l\in r} \frac{\lambda_r}{\mu_r} < c_l, \qquad \forall l. \tag{25}$$

What is less obvious is that this is also the sufficient condition for stability. In the rest of this section, we will prove that the condition (25) is sufficient for stochastic stability.

From our assumptions on the arrival process and file-size distribution, the evolution of $\mathbf{n}(\mathrm{t})$ can be described by a continuous-time Markov chain. It is more convenient to work with a discrete-time Markov chain and therefore, using uniformization [0], we consider the system only at certain event times, where the events correspond to arrival, departure and fictitious departures. Suppose that there are n_r files using route r immediately after an event. Then,

- with probability $\lambda_r / \sum_s (\lambda_s + \mu_s \zeta_s)$, the next event is an arrival to route r;
- with probability $\mu_r n_r x_r / \sum_s (\lambda_s + \mu_s \zeta_s)$, the next event is a departure from route r; and
- with probability $(\mu_r \zeta_r - \mu_r x_r n_r) / \sum_s (\lambda_s + \mu_s \zeta_s)$, the next event is a fictitious departure from route r.

Let $\mathbf{n}(k)$ denote the state of the Markov chain after the k^{th} event. Then $\mathbf{n}(k)$ is a discrete-time Markov chain. We are interested in showing that this Markov chain is positive chain. For this purpose, we will use the following result known as the Foster-Lyapunov criterion [0].

Theorem 2. *Suppose that a Markov chain* $\{\mathbf{n}(k)\}$ *is irreducible and let* E_0 *be a finite subset of the state space* E*. Then the chain is positive recurrent if, for some* $V : E \longrightarrow \Re$ *and some* $\delta > 0$*, we have* $\inf_{\mathbf{n}\in E} V(\mathbf{n}) > -\infty$ *and*

$$\begin{array}{ll} E(V(\mathbf{n}(k+1))|\mathbf{n}(k) = \mathbf{n}) < \infty, & \mathbf{n} \in E_0, \\ E(V(\mathbf{n}(k+1))|\mathbf{n}(k) = \mathbf{n}) \leq V(\mathbf{n}) - \delta, & \mathbf{n} \notin E_0. \end{array} \tag{26}$$

□

It is easy to verify that the Markov chain under consideration is irreducible. Given any state, with non-zero probability, it is possible to reach the zero state where $n_r = 0$ for all routes r. This will happen if there are no arrivals for all the routes for a sufficiently long period of time. From the zero state, it is possible to reach any state by considering a sequence of arrival events and no departure events, which again can happen with non-zero probability. Thus, one can reach any state from any other state in finite time with non-zero probability which, by definition, means that the Markov chain is irreducible. Now, to verify the key drift condition of the Foster-Lyapunov criterion, consider the candidate Lyapunov function

$$V(k) = \sum_r \kappa_r n_r^{\alpha+1}(k),$$

where the κ_r's are non-negative constants, to be chosen appropriately later. For notational convenience, and without loss of generality, we assume that

$$\sum_s (\lambda_s + \mu_s \zeta_s) = 1.$$

We note that

$$E(V(k+1) - V(k)|\mathbf{n}(k)) = \sum_r \lambda_r \kappa_r \left((n_r+1)^{\alpha+1} - n_r^{\alpha+1}\right) \tag{27}$$

$$+ \sum_r I_{n_r>0} \mu_r \hat{\chi}_r \kappa_r \left((n_r-1)^{\alpha+1} - n_r^{\alpha+1}\right) \tag{28}$$

$$= \sum_r \lambda_r \kappa_r \left((n_r+1)^{\alpha+1} - n_r^{\alpha+1}\right) I_{n_r>0} + \sum_r \lambda_r \kappa_r$$

$$+ \sum_r I_{n_r>0} \mu_r \hat{\chi}_r \kappa_r \left((n_r-1)^{\alpha+1} - n_r^{\alpha+1}\right)$$

Consider the expression

$$(n_r+1)^{\alpha+1} - n_r^{\alpha+1}$$

and assume $n_r > 0$. By Taylor's theorem,

$$(n_r+1)^{\alpha+1} - n_r^{\alpha+1} = (\alpha+1)n_r^{\alpha} + \frac{\alpha(\alpha+1)}{2}\tilde{n}_r^{\alpha-1}$$

for some $\tilde{n}_r$ that satisfies

$$n_r \leq \tilde{n}_r \leq n_r + 1.$$

Equivalently,

$$1 \leq \frac{\tilde{n}_r}{n_r} \leq 1 + \frac{1}{n_r} \leq 2.$$

Thus, for $\alpha \geq 1$,

$$\left(\frac{\tilde{n}_r}{n_r}\right)^{\alpha-1} \leq 2^{\alpha-1}$$

and, for $\alpha < 1$,

$$\left(\frac{\tilde{n}_r}{n_r}\right)^{\alpha-1} \leq 1.$$

Letting

$$C_1 = \frac{\alpha(\alpha+1)}{2} \max\{2^{\alpha-1}, 1\},$$

we have

$$(n_r + 1)^{\alpha+1} - n_r^{\alpha+1} \leq (\alpha+1)n_r^{\alpha} + C_1 n_r^{\alpha-1} \tag{29}$$

if $n_r > 0$.

Next, again by Taylor's theorem, we have

$$(n_r - 1)^{\alpha+1} - n_r^{\alpha+1} \leq -(\alpha+1)n_r^{\alpha} + \frac{\alpha(\alpha+1)}{2} m_r^{\alpha-1},$$

where m_r satisfies

$$n_r - 1 \leq m_r \leq n_r.$$

Assuming, $n_r \neq 0$, we have

$$1 - \frac{1}{n_r} \leq \frac{m_r}{n_r} \leq 1.$$

If $\alpha > 1$, then

$$\left(\frac{m_r}{n_r}\right)^{\alpha-1} \leq 1.$$

If $\alpha < 1$ and $n_r > 1$, then

$$\left(\frac{m_r}{n_r}\right)^{\alpha-1} \leq 2^{\alpha-1}$$

since $1 - 1/n_r \geq 1/2$. If $n_r = 1$, denote the corresponding value of m_r as $\tilde{m}$. Defining $C_2 = \frac{\alpha(\alpha+1)}{2} \max\{1, 2^{\alpha-1}, \tilde{m}^{\alpha-1}\}$, we have

$$(n_r - 1)^{\alpha+1} - n_r^{\alpha+1} \leq -(\alpha+1)n_r^{\alpha} + C_2 n_r^{\alpha-1}, \tag{30}$$

when $n_r > 0$. Substituting (29) and (30) in (29), we get

$$E(V(k+1)-V(k)|\mathbf{n}(k)) \leq \sum_r \kappa_r(\alpha+1)n_r^\alpha(\lambda_r-\mu_r\hat{\chi}_r) + C\sum_{n_r} I_{n_r>0}n_r^{\alpha-1} + \sum_r \lambda_r\kappa_r,$$

where $C = C_1 + C_2$.

We now make use of the following well-known result from concave optimization: suppose $\hat{\mathbf{y}}$ is the optimal solution to $\max_{\mathbf{y}} f(\mathbf{y})$ subject to $\mathbf{y} \in \mathcal{S}$, where $f(\cdot)$ is a concave function and $\mathcal{S}$ is a convex set. Then, for any $\mathbf{y} \in \mathcal{S}$, the following is true:

$$\sum_i \frac{\partial f}{\partial y_i}(\mathbf{y})(y_i - \hat{y}_i) \leq 0.$$

To make use of this result, let us rewrite the drift as

$$E(V(k+1)-V(k)|\mathbf{n}(k)) \leq \sum_r \kappa_r(\alpha+1)n_r^\alpha\mu(\frac{\lambda_r}{\mu_r}-\hat{\chi}_r) + C\sum_{n_r} I_{n_r>0}n_r^{\alpha-1} + \sum_r \lambda_r\kappa_r,$$

Recall that $\{\hat{\chi}_r\}$ is the optimal solution to (24). Further, if we choose $\chi_r = \frac{\lambda_r}{\mu_r}$, then this set of rates satisfy the link capacity constraints. In fact, from condition (25), there exists an $\varepsilon > 0$ such that

$$(1+\varepsilon)\sum_{r:l\in r}\frac{\lambda_r}{\mu_r} \leq c_l, \qquad \forall l.$$

Thus,

$$\chi_r = (1+\varepsilon)\frac{\lambda_r}{\mu_r}$$

is also a rate allocation that satisfies the link capacity constraints. Thus, from the property of concave optimization discussed earlier, we have

$$\sum_r \frac{n_r^\alpha w_r}{\frac{\lambda_r(1+\varepsilon)}{\mu_r}}\left(\frac{\lambda_r(1+\varepsilon)}{\mu_r}-\hat{\chi}_r\right) \leq 0. \tag{31}$$

This can be equivalently written as

$$\sum_r w_r n_r^\alpha\left(\frac{\mu_r}{\lambda_r}\right)^\alpha\left(\frac{\lambda_r}{\mu_r}-\hat{\chi}_r\right) \leq -\varepsilon\sum_r w_r n_r^\alpha\left(\frac{\mu_r}{\lambda_r}\right)^{\alpha-1}.$$

Choosing

$$\kappa_r = \frac{w_r}{(\alpha+1)\mu_r}\left(\frac{\mu_r}{\lambda}\right)^\alpha,$$

the drift can then be upper-bounded as

$$E(V(k+1)-V(k)|\mathbf{n}(k)) \leq -\varepsilon \sum_r w_r n_r^\alpha \left(\frac{\mu_r}{\lambda_r}\right)^{\alpha-1} + C\sum_{n_r} I_{n_r>0} n_r^{\alpha-1} + \sum_r \lambda_r \kappa_r.$$

Now it is easy to see that, given $\delta > 0$, there exists a $B > 0$ such that, for all $\mathbf{n}$ satisfying $\|\mathbf{n}\| > B$,

$$E(V(k+1)-V(k)|\mathbf{n}(k)) \leq -\delta.$$

Thus, by the Foster-Lyapunov criterion, the Markov chain describing the number of files waiting on the various routes in the network is positive recurrent.

In addition to showing positive recurrence, it is also possible to obtain an upper bound on the steady-state value of $E(\|\mathbf{n}\|^\alpha)$. To obtain such a bound, first note that

$$n_r \leq \|\mathbf{n}\|, \qquad \forall r,$$

and there exists j such that

$$n_j \geq \frac{\|\mathbf{n}\|}{R},$$

where R is the dimension of $\mathbf{n}$, i.e., R is the total number of routes in the network. Thus, for appropriately chosen positive constants K_1 and K, the drift can be bounded as

$$E(V(k+1)-V(k)|\mathbf{n}(k)) \leq -\varepsilon K_1 \|\mathbf{n}(k)\|^\alpha + C\|\mathbf{n}(k)\|^{\alpha-1} \sum_r I_{n_r(k)>0} + K.$$

Since

$$\frac{1}{\|\mathbf{n}(k)\|} \sum_r I_{n_r(k)>0} \leq \frac{R}{\|\mathbf{n}\|},$$

we have

$$\begin{aligned} E(V(k+1)-V(k)|\mathbf{n}(k)) &\leq -\varepsilon K_1 \|\mathbf{n}(k)\|^\alpha + RC\|\mathbf{n}(k)\|^{\alpha-1} I_{\|\mathbf{n}\|\geq B} \\ &\quad + C\|\mathbf{n}(k)\|^{\alpha-1} I_{\{1\leq\|\mathbf{n}\|<B\}} + K \\ &\leq -\varepsilon K_1 \|\mathbf{n}(k)\|^\alpha + \frac{RC}{B}\|\mathbf{n}(k)\|^\alpha \\ &\quad + RC\max\{1, B^{\alpha-1}\} + K, \end{aligned}$$

where B is an arbitrary positive constant that is chosen, depending on ϵ, to ensure that

$$\frac{RC}{B} - \varepsilon K_1 < 0.$$

Thus,

$$(\varepsilon K_1 - K_2) E\left(\|\mathbf{n}(k)\|^\alpha\right) \leq E(V(k) - V(k+1)) + K_3,$$

where

$$K_2 = \frac{RC}{B}, \qquad K_3 = K + RC \max\{1, B^{\alpha-1}\}.$$

Summing over k, we get

$$\sum_{k=1}^{M} (\varepsilon K_1 - K_2) E\left(\|\mathbf{n}(k)\|^{\alpha}\right) \leq E(V(1)) - E(V(M+1)) + MK_3$$
$$\leq E(V(1)) + MK_3.$$

Finally,

$$\limsup_{M\to\infty} \frac{1}{M} \sum_{k=1}^{M} E\left(\|\mathbf{n}(k)\|^{\alpha}\right) \leq \frac{K_3}{\varepsilon K_1 - K_2}.$$

Since the Markov chain is ergodic, the bound also applies to the steady-state α^{th} moment. Note that, as the load on any link gets close to the link capacity, ϵ decreases and the corresponding value of B required to ensure that $\varepsilon K_1 - K_2$ is positive increases. In fact, this means that $\varepsilon K_1 - K_2$ decreases and therefore, the bound on the α^{th} moment increases, which is reasonable when the system load increases.

The above result has been obtained under the assumption that the file arrival processes are Poisson and the file-size distributions are exponential. There is experimental evidence which suggests that the file-arrival process in the Internet is indeed Poisson. On the other hand, the file-size distribution is not as easily characterized. It is well-known that the file-sizes in the Internet have a large variance. However, the files that we consider here are only those that are large enough so that it reasonable to assume that compared to the file arrival and departure time scale, the congestion-control algorithms that control the file transfers converge to the solution of (24). It is difficult to characterize the file-size distribution of such files alone. Thus, it is of much interest to extend the above stability result to general file-size distributions.

Extending the result to hyper-exponential file-size distributions is immediate. Suppose that the file-size distribution on the r^{th} route is exponential with mean $1/\mu_{ri}$ with probability p_{ri}, for $i = 1, , 2, \ldots I(r)$ for some $I(r) > 1$. Then, the arrival process to the r^{th} route can be thought as $I(r)$ independent Poisson processes with mean rates $\lambda_r p_{ri}$, and the i^{th} such process generating files from an exponential distribution with mean $1/\mu_{ri}$. This is no different from the stability problem that we have already considered and therefore, the Markov chain describing the number of files in the network is still positive recurrent. A more complicated extension is for the case where the resource allocation parameter α is chosen to be equal to 1 and the weights w_r are all equal to 1, i.e., $U_r(x_r) = \log x_r$, $\forall r$. For this special case and for certain network topologies, using a reversibility argument, it has been shown in [0, 0, 0] that the connection-level system has a steady-state distribution which is independent of the file-size distribution. Extensions to general file-size distributions, general topologies and general resource allocation parameters is still an open problem. For simple topologies and Coxian file-size distributions with a small

number of phases, numerical construction of Lyapunov functions suggests that the stability condition may be the same irrespective of the file-size distribution [0].

6 Conclusions

In this paper, we have surveyed recent research on the topic of Internet congestion control. We have emphasized the role of optimization, control theory and stochastic models in developing algorithms that perform well even in the presence of delays and stochastic disturbances such as arrivals and departures of flows.

Fairness and stability are two important considerations in designing congestion control mechanisms that allow resource-sharing in a network with many competing users. By associating a utility function with each user and defining fairness to mean the maximization of the sum utility of the network, the fair resource sharing problem can be viewed as an optimization problem. In the language of convex optimization, the congestion controllers can be thought as numerical techniques (such as gradient search) to solve an optimization problem whose objective is social welfare. The congestion feedback from the network can be thought of as shadow prices or Lagrange multipliers computed by the links in the network to ensure that each link in the network is not over-utilized.

Designing the step sizes for the gradient procedure is difficult since the controllers that implement these algorithms are distributed in the network. Each controller's congestion feedback from the network is subject to propagation delay, and possibly queueing delay, and the challenge is to design the control gains (which are the step sizes of the gradient search procedure) to ensure stability in the presence of heterogeneous feedback delays. We used Razumikhin's theory to perform the controller design.

The randomness in the network due to file arrivals and departures was captured using a Markov chain model at the connection level. We then used stochastic Lyapunov analysis to show that the resource allocation formulation leads to efficient network operation.

Acknowledgments:

It is a pleasure to thank Ashvin Lakshmikantha for providing useful comments on an earlier draft of this paper.

References

1. T. Alpcan and T. Başar (2003) *A utility-based congestion control scheme for internet-style networks with delay*, Proceedings of IEEE Infocom (San Francisco, California)

2. S. Asmussen (2003) *Applied Probability and Queues*, Springer-Verlag, New York
3. D. Bertsekas (1995) *Nonlinear programming*, Athena Scientific, Belmont, MA, 1995
4. T. Bonald and L. Massoulie (2001) *Impact of fairness on Internet performance*, Proceedings of ACM Sigmetrics
5. T. Bonald and A. Proutiere (2003) *Insensitive bandwidth sharing in data networks*, Queueing Systems **44**, 69–100
6. G. de Veciana, T.-J. Lee, and T. Konstantopoulos (2000) *Stability and performance analysis of networks supporting elastic services*, IEEE/ACM Transactions on Networking **9** (2001), no. 1, 2–14
7. J. Hale and S. M. V. Lunel (1991) *Introduction to functional differential equations*, 2nd edition, Springer Verlag, New York, NY
8. C.V. Hollot, V. Misra, D. Towsley, and W. Gong (2001) *On designing improved controllers for AQM routers supporting TCP flows*, Proceedings of IEEE Infocom (Anchorage, Alaska, pp. 1726–1734
9. V. Jacobson (1988) *Congestion avoidance and control*, ACM Computer Communication Review **18**, 314–329
10. R. Johari and D. Tan (2001) *End-to-end congestion control for the Internet: Delays and stability*, IEEE/ACM Transactions on Networking **9**, no. 6, 818–832
11. F. P. Kelly (1997) *Charging and rate control for elastic traffic*, European Transactions on Telecommunications **8** (1997), 33–37
12. F. P. Kelly (2003) *Fairness and stability of end-to-end congestion control*, European Journal of Control **9**, 149–165
13. F. P. Kelly, A. Maulloo, and D. Tan (1998) *Rate control in communication networks: shadow prices, proportional fairness and stability*, Journal of the Operational Research Society **49**, 237–252
14. P.R. Kumar and P. Varaiya (1986) *Stochastic Systems, Estimation, Identification and Adaptive Control*, Prentice Hall, 1986
15. S. Kunniyur and R. Srikant (2000) *End-to-end congestion control: utility functions, random losses and ECN marks*, Proceedings of IEEE Infocom (Tel Aviv, Israel)
16. S. Kunniyur and R. Srikant (2003) *End-to-end congestion control: utility functions, random losses and ECN marks*, IEEE/ACM Transactions on Networking **7**, no. 5, 689–702
17. S. Kunniyur and R. Srikant (2003) *Stable, scalable, fair congestion control and AQM schemes that achieve high utilization in the internet*, IEEE Transactions on Automatic Control **48**, 2024–2028
18. R. La, P. Ranjan, and E. Abed (2004) *Global stability conditions for rate control with arbitrary communication delays*, University of Maryland CSHCN TR 2003-25
19. A. Lakshmikantha, C. Beck, and R. Srikant (2004) *Connection level analysis of the Internet using the sum of squares technique*, Proceedings of the Conference on Information Sciences and Systems (Princeton, NJ)
20. S. Liu, T. Başar, and R. Srikant (2003) *Exponential RED: A stabilizing AQM scheme for low and high speed TCP*, University of Illinois Tech Report. An earlier version appeared in the Proceedings of IEEE Conference on Decision and Control, December 2003, under the title "Controlling the Internet: A survey and some new results"
21. S. H. Low and D. E. Lapsley (1999) *Optimization flow control, I: Basic algorithm and convergence*, IEEE/ACM Transactions on Networking (1999), 861–875

22. L. Massoulie (2002) *Stability of distributed congestion control with heterogenous feedback delays*, IEEE Transactions on Automatic Control **47** (2002), 895–902
23. V. Misra, W. Gong, and D. Towsley (2000) *A fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED*, Proceedings of ACM Sigcomm (Stockholm, Sweden)
24. F. Paganini, J. Doyle, and S. Low (2000) *Scalable laws for stable network congestion control*, Proceedings of the IEEE Conference on Decision and Control (Orlando, FL), pp. 185–190.
25. G. Raina (2003) *Personal communication*
26. J. Roberts and L. Massoulie (1998) *Bandwidth sharing and admission control for elastic traffic*, Proc. ITC specialists seminar (Yokohama, Japan)
27. R. Srikant (2004) *The Mathematics of Internet Congestion Control*, Birkhauser
28. G. Vinnicombe (2001) *On the stability of end-to-end congestion control for the Internet*, University of Cambridge Technical Report CUED/F-INFENG/TR.398. Available at http://www.eng.cam.ac.uk/˜gv
29. G. Vinnicombe (2002) *On the stability of networks operating TCP-like congestion control*, Proceedings of the IFAC World Congress (Barcelona, Spain)
30. J.T. Wen and M. Arcak (2003) *A unifying passivity framework for network flow control*, Proceedings of IEEE Infocom
31. H. Yaiche, R. R. Mazumdar, and C. Rosenberg (2000) *A game-theoretic framework for bandwidth allocation and pricing in broadband networks*, IEEE/ACM Transactions on Networking **8**, no. 5, 667–678.
32. L. Ying, G. Dullerud, and R. Srikant (2004) *Global stability of Internet congestion controllers with heterogeneous delays*, To appear in the Proceedings of the American Control Conference (Boston, MA)