

# TCP-Illinois: A Loss and Delay-Based Congestion Control Algorithm for High-Speed Networks

Shao Liu, Tamer Başar and R. Srikant

**Abstract**— We introduce a new congestion control algorithm, called TCP-Illinois, which has many desirable properties for implementation in (very) high-speed networks. TCP-Illinois is a sender side protocol, which modifies the AIMD algorithm of the standard TCP (Reno, NewReno or SACK) by adjusting the increment/decrement amounts based on delay information. By using both loss and delay as congestion signals, TCP-Illinois achieves a better throughput than the standard TCP for high-speed networks. To study its fairness and stability properties, we extend recently developed stochastic matrix models of TCP to accommodate window size backoff probabilities that are proportional to arrival rates when the network is congested. Using this model, TCP-Illinois is shown to allocate the network resource fairly as in the standard TCP. In addition, TCP-Illinois is shown to be compatible with the standard TCP when implemented in today’s networks, and is shown to provide the right incentive for transition to the new protocol. We finally perform *ns-2* simulations to validate its properties and demonstrate its performance.

**Keywords:** Congestion Control, TCP, Reno, NewReno, SACK, AIMD, packet loss, queuing delay, throughput, fairness, stability

## I. INTRODUCTION

TCP-Reno [17], TCP-NewReno [12], and SACK TCP [25] are the standard versions of TCP congestion control protocols currently deployed in the Internet, and they have achieved great success in performing congestion avoidance and control. The key feature for the standard TCP is its congestion avoidance phase, which uses the additive increment multiplicative decrement (AIMD) algorithm [16]. Being a window-based algorithm, TCP controls the send rate by maintaining a window size variable  $W$ , which limits the number of unacknowledged packets in the network from a single user. This window size is adjusted by the AIMD algorithm in the following manner:  $W$  is increased by  $\alpha/W$  ( $\alpha = 1$  for standard setting) for each ACK, and thus is increased by a constant  $\alpha/b$  per round trip time (RTT) if all the packets are acknowledged within an RTT, where  $b$  is the number of packets acknowledged by each ACK ( $b = 1$  for original TCP, and  $b = 2$  for delayed ACK [30]). On the other hand,  $W$  is decreased by a fixed proportion  $\beta W$  ( $\beta = 1/2$  for standard setting) once some packets are detected to be lost in the last RTT<sup>1</sup>. Under this algorithm, senders

gently probe the network for spare bandwidth by cautiously increasing their send rates, and sharply reduce their send rates when congestion is detected. Along with other features like slow start, fast recovery, and fast retransmission, TCP achieves congestion control successfully in the current low speed networks.

However, the current TCP can perform poorly in networks with high bandwidth-delay product (BDP) paths, since the AIMD algorithm, being very conservative, is not designed for large window size flows. First, it takes too long time for a large window size user to recover after a backoff and the bandwidth is not effectively utilized [11]. Second, TCP’s time average window size  $\bar{W}$  is related with the loss event probability<sup>2</sup>  $p$  in the following manner [27]

$$W \approx \sqrt{3/2bp} \quad \text{or} \quad p \approx 3/2bW^2. \quad (1)$$

Since TCP interprets all packet losses as congestion signals,  $W$  is upper bounded by  $\sqrt{3/2bp_t}$ , where  $p_t$  is the transmission error rate [11].  $p_t$  is around  $10^{-7}$  in optical fiber networks, and even higher in other lossy networks, like wireless networks. So TCP, and its AIMD algorithm in particular, should be modified in high bandwidth delay product networks.

Several alternatives to current versions of TCP have been proposed for implementation in high-speed networks. Some require the modification to router algorithms also, like XCP [19], and some modify the sender side only, like HS-TCP [11], Scalable TCP [20], TCP-Westwood [35], H-TCP [22], BIC-TCP [34], TCP Vegas [9] and FAST TCP [18]. Although each of these has shown advantages over TCP in some aspects, none of them have yet provided convincing evidence that they are overwhelmingly better than TCP and are suitable for general deployment. In this paper, we first list some desirable design specifications that a high speed TCP variant should meet, and then introduce and analyze a new TCP variant called TCP-Illinois, which satisfies all the requirements and outperforms the current TCP and some other variants.

## II. BACKGROUND AND MOTIVATION

As we have mentioned above, several new protocols have been introduced to replace the standard TCP in high speed networks. To compare these protocols and to provide insight into the development of an ideal protocol, we list below some requirements that a new protocol should satisfy. This list

All the three authors are with the Department of Electrical and Computer Engineering and Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, 1308 West Main Street, Urbana, IL 61801-2307, USA. Emails: (shaoliu,basar1,rsrikant)@uiuc.edu

Research supported by the NSF ITR Grant CCR 00-85917, DARPA Grant F30602-00-2-0542 and AFOSR URI F49620-01-1-0365.

<sup>1</sup>Within one RTT,  $W$  may decrease multiple times in Reno and can decrease only once in NewReno and SACK.

<sup>2</sup>All the losses within one RTT are regarded as one loss event. Loss event probability is the number of loss events divided by the number of packets sent.

broadens and makes more complete, a list of requirements that can be found in [22].

**Intra-protocol requirements:** The requirements that the protocol should satisfy in a network consisting of a single protocol are the following:

- 1) **Efficiency.** The average throughput for the new protocol should be larger than that of TCP-Reno in high speed networks.
- 2) **Intra-Protocol Fairness.** Network resources should be fairly allocated to all flows. The fairness here does not necessarily mean that all flows sharing the same link achieve the same throughput. Instead, this means that the new protocol should not be significantly more unfair than the current TCP. For example, under the current TCP, flows with different RTTs achieve similar average window sizes, and their average throughput are inversely proportional to their RTTs. The new protocol should not be significantly more biased against long RTT flows.
- 3) **Stability and Responsiveness.** The congestion control system formed by the new protocol should converge to the unique equilibrium state quickly, starting from any initial state.
- 4) **Heavy Congestion<sup>3</sup> Avoidance.** A simple idea to achieve a larger average window size for a given loss event probability is to choose a larger value for  $\alpha$  and a smaller value for  $\beta$ . However, a faster increase and a smaller decrement may cause heavy congestion more frequently, and thus lead to some undesirable or even catastrophe consequences. First, heavy congestion causes more timeout and makes TCP enter slow start phase more often, and causes under-utilization. For example, HS-TCP, as to be demonstrated later, faces timeout regularly if SACK is not used. Second, heavy congestion causes synchronization more often, which makes the resource allocation very unfair for large RTT users, as will be discussed later. Finally, if all the users choose the more aggressive policy, congestion collapse is more likely to happen. By the consideration of this requirement, we should not pick  $\alpha$  and  $\beta$  values in a manner that leads to behavior which may be counter-productive.
- 5) **Router Independence.** The new protocol should work well regardless of the router condition, like the buffer size of the router, and the queue management algorithm of the router (Droptail or some Active Queue Management (AQM) schemes, like RED [13]). With a more advanced router, like with a larger buffer or an AQM support, the new protocol might achieve better performance, but the performance with Droptail and small buffer should be good also.
- 6) **Performance in Wireless Networks.** The new proto-

<sup>3</sup>In our context, heavy congestion means that many packets are dropped when congestion happens. It only concerns the time when congestion happens and it does not necessarily mean that the packet loss probability or the loss event probability is high. In some other papers, it is called heavy synchronization.

col is mainly for wired networks, but it is a plus if it also performs well in wireless networks and other lossy networks.

**Inter-protocol requirements:** The requirements on the protocol when it co-exists in a network with the standard TCP are the following:

- 1) **Compatibility.** In low speed networks, the new protocol should achieve a similar rate to that of the standard TCP; and in high-speed networks, the standard TCP should not suffer significant throughput loss.
- 2) **Incentive to switch.** By switching to the new protocol from the standard TCP, the users should get a larger average throughput.

In the above list, the wireless network performance requirement is desirable but not essential, whereas all others are essential.

We now briefly discuss existing TCP variants to see whether they satisfy all these requirements. First, it is impractical to modify routers if the benefit is marginal or can be achieved by sender side modifications, and thus algorithms which need router side modifications, like XCP, are not ideal. Without modifying the router, a sender has only two congestion signals: packet loss and queueing delay. We can thus classify all the sender-side protocols into one of two classes. Loss based congestion avoidance (LCA) algorithms, like HS-TCP and Scalable TCP, use packet loss as primary congestion signal, increase window size for each ACK and decrease window size for packet loss. LCA algorithms can be regarded as general AIMD algorithms, since the only difference from AIMD is that they set different  $\alpha$  and  $\beta$  values and allow them to be variables. On the other hand, delay based congestion avoidance (DCA) algorithms, like TCP-Vegas and FAST TCP, are fundamentally different from AIMD, as they use queueing delay as the primary congestion signal, increase window size if delay is small and decrease window size if delay is large.

The advantage of DCA algorithms is that they achieve better average throughput, since they can keep the system around full utilization. As a comparison, the LCA algorithms purposely generate packet losses and oscillate between full utilization and under utilization. However, existing DCA algorithms suffer from inherent weaknesses. First, they are not compatible with the standard TCP. TCP-Vegas gets a very small share of the link capacity if competing with TCP-Reno [26], [1]; and FAST TCP yields non-unique equilibrium point if competing with TCP-Reno: the allocation of the bandwidth between FAST and Reno users depend on which users enter the network first [32]. Second, they require the buffer size at the router to be larger than a specified value and this value increases with the number of users  $N$ . Both Vegas and FAST control the number of packets queued in the router for each flow, and this number cannot be too small. The requirement for the router buffer is thus  $N$  times this number. For a fixed buffer size, there is an upper bound on  $N$  for Vegas or FAST to work functionally. Finally and most important, the performance of all these DCA algorithms highly depend

on the accurate measurement of the queueing delays. In the real networks, RTT is the sum of propagation delay, queueing delay and a random noise term, which comes from various reasons like scheduling delay. The measured queueing delay is thus the sum of the real queueing delay and this random noise. As the mean or the variation of this random noise becomes large, the correlation of increased RTT and packet loss becomes weak, and thus the performance of these DCA algorithms is downgraded significantly and sometimes even completely fail to work, as shown in Section VI-D. This major problem of DCA algorithms raises doubts on whether DCA algorithms would be effective in practice; see [8], [24], [28] for some discussions.

On the other hand, none of the existing loss based algorithms satisfy all the requirements. Scalable TCP sets  $\alpha$  proportional to  $W$ , but it has been demonstrated to be unfair (see [22], Fig. 2). HS-TCP sets  $\alpha$  to be a step-wise increasing function of  $W$ , and  $\beta$  a step-wise decreasing function of  $W$ , but its convergence speed is very slow (see [22], Fig. 1). H-TCP aims at a faster convergence and better utilization by setting  $\alpha$  to be an increasing function of the time elapsed since last backoff and setting  $\beta$  to be such that the link is always around full utilization, even after the backoff. For all the above algorithms, the increase is initially slow, when the window size is small and the network is far from congestion, but becomes fast later, when the window size is large and the network is close to congestion. As a result, the window size curve between two consecutive loss events is convex; see bottom plot of Fig. 5. This convex nature is not desired. First, the slow increment when the network is far from congestion is inefficient. For a given  $\beta$ , the convex window curve gets an even smaller average throughput than traditional linear increase, and thus these algorithms have to choose a smaller  $\beta < 1/2$ , which is not fair with the standard TCP. Second, the fast increment when the network is close to congestion causes heavy congestion more easily. As we have mentioned before and will further discuss later, heavy congestion causes more frequent timeouts, more synchronized sender backoff, and is unfair to large RTT users and the standard TCP users. In summary, the main problem with existing general AIMD algorithms is the convexity of the  $W$  curve. An ideal window curve should be concave, which is more efficient and avoids heavy congestion. An objective of our work is to design a general AIMD algorithm which results in a concave curve. All algorithms with a concave window curve will be called Concave-AIMD algorithms (CAIMD).

### III. LDCA-CAIMD ALGORITHMS AND THE TCP-ILLINOIS PROTOCOL

To achieve the concave curve, we should set  $\alpha$  large when far from congestion and set it small when close to congestion. To achieve a better throughput in networks with packet losses not due to congestion and to be fair with the standard TCP, we should also set  $\beta$  small when far from congestion and set it large when close to congestion. The difficulty is in judging whether the congestion is imminent or not, since it requires an estimation of the current congestion level. Before

congestion (packet loss) really happens, the only congestion indicating information is queueing delay. So our key idea is that: when the average queueing delay  $d_a$  is small, the sender predicts that the congestion is not imminent and sets a large  $\alpha$  and small  $\beta$ ; when  $d_a$  is large, the sender predicts that the congestion is imminent and sets a small  $\alpha$  and large  $\beta$ . As a result,  $\alpha = f_1(d_a)$  and  $\beta = f_2(d_a)$ , where  $f_1(\cdot)$  is decreasing and  $f_2(\cdot)$  is increasing. Any combination of increasing  $f_1(\cdot)$  and decreasing  $f_2(\cdot)$  functions results in a CAIMD algorithms.

From this key idea, both loss and delay are considered in congestion avoidance, and this idea leads to a class of loss-delay based congestion avoidance (LDCA) algorithms, which we call LDCA-CAIMD. These LDCA-CAIMD algorithms use loss to determine the *direction* and use delay to adjust the *pace* of window size change. So loss is the primary congestion signal and delay is an assistant congestion signal. This makes LDCA-CAIMD fundamentally different from all DCA algorithms and some other LDCA algorithms, like TCP-Compound [31], which use delay to determine the *direction* (maybe *pace* also) of window size change. Therefore, all previous algorithms which use delay as congestion signal treat delay as the primary congestion signal. As we have mentioned, a big problem using delay to control congestion is that delay cannot be measured accurately and usually the RTT measurement is buried with noise. If delay determines the *direction* of window size change, noisy RTT measurement degrades the performance significant, since it could be very often that window size actually increases (respectively, decreases) when it should decrease (respectively, increase). As a major advantage, our LDCA-CAIMD algorithms, using delay only as an assistant signal, are much more robust to noise in RTT measurements, as discussed in VI-D.

There are numerous choices of  $f_1(\cdot)$  and  $f_2(\cdot)$ . An example is as below:

$$\alpha = f_1(d_a) = \begin{cases} \alpha_{max} & \text{if } d_a \leq d_1 \\ \frac{\kappa_1}{\kappa_2 + d_a} & \text{otherwise.} \end{cases} \quad (2)$$

$$\beta = f_2(d_a) = \begin{cases} \beta_{min} & \text{if } d_a \leq d_2 \\ \kappa_3 + \kappa_4 d_a & \text{if } d_2 < d_a < d_3 \\ \beta_{max} & \text{otherwise.} \end{cases} \quad (3)$$

We let  $f_1(\cdot)$  and  $f_2(\cdot)$  be continuous functions and thus  $\frac{\kappa_1}{\kappa_2 + d_1} = \alpha_{max}$ ,  $\beta_{min} = \kappa_3 + \kappa_4 d_2$  and  $\beta_{max} = \kappa_3 + \kappa_4 d_3$ . Suppose  $d_m$  is the maximum average queueing delay and we denote  $\alpha_{min} = f_1(d_m)$ , then we also have  $\frac{\kappa_1}{\kappa_2 + d_m} = \alpha_{min}$ . From these conditions, we have

$$\begin{aligned} \kappa_1 &= \frac{(d_m - d_1)\alpha_{min}\alpha_{max}}{\alpha_{max} - \alpha_{min}} & \text{and} & \quad \kappa_2 = \frac{(d_m - d_1)\alpha_{min}}{\alpha_{max} - \alpha_{min}} - d_1, \\ \kappa_3 &= \frac{\beta_{min}d_3 - \beta_{max}d_2}{d_3 - d_2} & \text{and} & \quad \kappa_4 = \frac{\beta_{max} - \beta_{min}}{d_3 - d_2}. \end{aligned} \quad (4)$$

This choice of  $f_1(\cdot)$  and  $f_2(\cdot)$  is demonstrated in Fig. 1. According to this choice, we now design a specific LDCA-CAIMD protocol, which we call TCP-Illinois. The protocol is described as below:

- All the features of TCP-NewReno except the AIMD algorithm are retained.

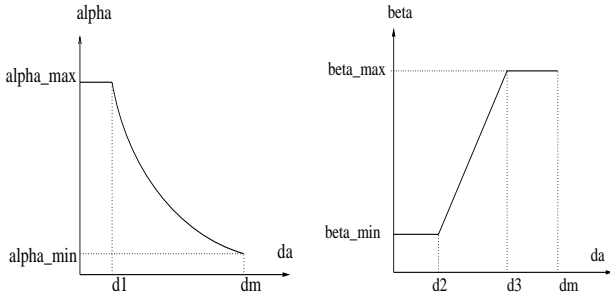


Fig. 1.  $\alpha$  and  $\beta$  curves Vs  $d_a$ .

- In the congestion avoidance phase, the sender measures RTT  $T$  for each Ack, and average the RTT measurement over the last  $W$  acknowledgements (one RTT interval) to derive the average RTT  $T_a$ . The sender records the maximum and minimum average RTT ever seen as  $T_{max}$  and  $T_{min}$ , respectively, and computes the maximum average queueing delay  $d_m = T_{max} - T_{min}$  and the current average queueing delay  $d_a = T_a - T_{min}$ .
- The sender picks eight parameters  $0 < \alpha_{min} \leq 1 \leq \alpha_{max}$ ,  $0 < \beta_{min} \leq \beta_{max} \leq 1/2$ ,  $W_{thresh} > 0$ ,  $0 \leq \eta_1 < 1$ , and  $0 \leq \eta_2 \leq \eta_3 \leq 1$ . The sender sets  $d_i = \eta_i d_m$  ( $i = 1, 2, 3$ ), computes  $\kappa_i$  ( $i = 1, 2, 3, 4$ ) from (4), and computes the current  $\alpha$  and  $\beta$  values from (2) and (3). The  $\kappa_i$  ( $i = 1, 2, 3, 4$ ) values are updated if  $T_{max}$  or  $T_{min}$  is updated. The  $\alpha$  and  $\beta$  values are updated once per RTT.
- $\alpha \leftarrow 1$  and  $\beta \leftarrow 1/2$  if  $W < W_{thresh}$ .
- $W \leftarrow W + \alpha/W$  for each ACK.
- $W \leftarrow W - \beta W$ , if in the last RTT there is packet loss detected through triple duplicate ACK.
- Once there is a timeout, the sender sets the slow start threshold to be  $W/2$ , enters slow start phase, and resets  $\alpha = 1$  and  $\beta = 1/2$ , and  $\alpha$  and  $\beta$  values are unchanged until one RTT after the slow start phase ends.

TCP-Illinois retains the fast recover and fast retransmission features of NewReno in standard option. If the receivers support selective acknowledgement, TCP-Illinois can also back off its window size when packet loss is detected through select ACK and adopt features from SACK TCP. However, the SACK support is not needed, since TCP-Illinois avoids heavy congestion effectively.

#### IV. FAIRNESS AND STABILITY

In this section, we study the fairness of TCP-Illinois. This involves the intra-protocol fairness between different TCP-Illinois users and also the compatibility problem with TCP-Reno: the resource allocation between TCP-Illinois users and TCP-Reno users. We first develop a new stochastic matrix model of general AIMD algorithms and then study the fairness and stability properties under this new model.

##### A. Stochastic Matrix Model for General AIMD Algorithms

There have been several recent papers on the stochastic matrix model of general AIMD algorithms; see [3]–[5], [21], [29], [33]. We first provide an overview of this model using different notation and derivation from the corresponding

references, and then extend this model by modifying one major assumption of the former literatures. Throughout, we consider the single link case, analyze the congestion avoidance phase only.

Suppose a link with capacity  $C$  and queue limit  $B$  is shared by  $N$  users, indexed by  $i$  ( $i = 1, 2, \dots, N$ ). User  $i$  has window size  $W_i$ , window increment  $\alpha_i$ , backoff factor  $\beta_i$ , rate  $x_i$ , loss event probability  $p_i$ , RTT  $T_i = T_i^p + d_i$ , where  $T_i^p$  and  $d_i$  are the propagation and queueing delays, respectively. We define  $\mathbf{W} := [W_1, \dots, W_N]^T$ ,  $\mathbf{x} := [x_1, \dots, x_N]^T$ , and  $\beta[k] = [\beta_1[k], \dots, \beta_N[k]]^T$ . When the link is congested and one or more packets are dropped, we call this a congestion event, and denote by  $t_k$  the time at the  $k$ -th congestion event. At one congestion event, one or several or all flows see packet losses and backoff their window sizes, and we say that a loss event<sup>4</sup> happens for these flows. For any time varying variable  $v(t)$ , we use  $E[v]$  to denote its expected value, and use  $v[k]$  (respectively,  $v[k^+]$ ) to denote its value just before (respectively, after) the  $k$ -th congestion event. Here,  $v$  could stand for  $W_i$ ,  $x_i$ ,  $T_i$ ,  $\alpha_i$ ,  $\beta_i$ ,  $\mathbf{W}$ ,  $\mathbf{x}$ ,  $\beta$ , as well as some other variables to be introduced later.

At each congestion event  $k$ , the total arrival rate  $\sum_{i=1}^N x_i[k]$  is slightly larger than or approximately equals to the link capacity  $C$ . So the following relationship holds approximately:

$$\sum_{i=1}^N x_i[k] = C', \quad \forall k \in \mathbb{N} \cup \{0\}, \quad (5)$$

where  $C' \geq C$  is a constant. We define  $\Sigma = \{\mathbf{z} = [z_1, \dots, z_N]^T \in \mathbb{R}^N : z_i \geq 0, \sum_{i=1}^N z_i = C'\}$ , then  $\Sigma$  is the set of all the possible  $\mathbf{x}[k]$ 's, and we call  $\Sigma$  the feasible set of  $\mathbf{x}[k]$ .

Between two consecutive congestion events,  $W_i(t)$  is increased with rate  $\alpha_i(t)/T_i(t)$ , and thus

$$W_i[k+1] = W_i[k^+] + \int_{t_k}^{t_{k+1}} \frac{\alpha_i(t)}{T_i(t)} dt. \quad (6)$$

If we define

$$\tilde{T}_i[k] := \frac{\int_{t_k}^{t_{k+1}} \alpha(t) dt}{\int_{t_k}^{t_{k+1}} \frac{\alpha(t)}{T_i(t)} dt}. \quad (7)$$

Then, we have

$$W_i[k+1] = W_i[k^+] + \frac{1}{\tilde{T}_i[k]} \int_{t_k}^{t_{k+1}} \alpha_i(t) dt. \quad (8)$$

At each congestion event  $k$  and for each flow  $i$ , we define its loss event random variable  $E_i[k]$ :

$$E_i[k] := \begin{cases} 1 & \text{if flow } i \text{ sees at least one packet loss,} \\ 0 & \text{otherwise,} \end{cases} \quad (9)$$

and define  $\mathbf{E}[k] := [E_1[k], \dots, E_N[k]]^T$ . Note that  $E_i[k]$  and  $E_j[k]$  are correlated for all  $i \neq j$ , since  $\sum_{i=1}^N E_i[k] \geq 1$ .  $\mathbf{E}[k]$  takes its value from the set  $\mathbb{E} := (\mathbf{E}^{(1)}, \mathbf{E}^{(2)}, \dots, \mathbf{E}^{(\mu)})$ , where  $\mu = 2^N - 1$ , and  $\mathbf{E}^{(l)}$  ( $1 \leq l \leq \mu$ ) flips 1 to  $N$  entries of  $\mathbf{0} := [0, \dots, 0]^T$  from 0 to 1 (there are altogether  $\mu$  such flips). Define  $\rho_l[k] := \text{Prob}(\mathbf{E}[k] = \mathbf{E}^{(l)})$ , and  $q_l[k] := \text{Prob}(E_i[k] =$

<sup>4</sup>In our terminology, a congestion event is for a link, while a loss event is for an individual user.

1). Then  $\sum_{i=1}^{\mu} \rho_i[k] = 1$ , and  $q_i[k] = \sum_{l: E_i^{(l)}=1} \rho_l$ . Here,  $q_i[k]$  is the probability that flow  $i$  experiences a loss event and backoffs its window size at the  $k$ -th congestion event. Note that  $q_i[k]$  is different from loss event probability  $p_i$ , which is the total number of loss events divided by the total number of packets transmitted. To make this difference clear, we call  $q_i[k]$  flow  $i$ 's backoff probability at congestion event  $k$ .

With the loss event random variables defined, we have

$$W_i[k^+] = W_i[k](1 - \beta_i[k]E_i[k]). \quad (10)$$

Combining (8) and (10), and considering the relationship that  $x_i[k] = W_i[k]/T_i[k]$ , we have

$$x_i[k+1] = x_i[k](1 - \beta_i[k]E_i[k]) + \frac{1}{T_i[k]\tilde{T}_i[k]} \int_{t_k}^{t_{k+1}} \alpha_i(t) dt. \quad (11)$$

Equations (5) and (11) characterize the discrete-time stochastic system of any general AIMD algorithms.

### B. Markov Chain for Identical $\alpha_i(t)$

For the standard TCP,  $\alpha_i(t) \equiv 1$ ; for TCP-Illinois,  $\alpha_i(t)$  is the same for all users, since all users see the same queueing delay. For both of these algorithms and some other general AIMD algorithms, we have  $\alpha_i(t) = \alpha(t), \forall i \in \{1, 2, \dots, N\}, \forall t$ . For these algorithms, we have

$$\int_{t_k}^{t_{k+1}} \alpha(t) dt \sum_{i=1}^N (\tilde{T}_i[k]T_i[k])^{-1} = \sum_{i=1}^N \beta_i[k]E_i[k]x_i[k], \quad (12)$$

and thus

$$\int_{t_k}^{t_{k+1}} \alpha(t) dt = \frac{1}{\sum_{i=1}^N (T_i[k]\tilde{T}_i[k])^{-1}} \sum_{i=1}^N \beta_i[k]E_i[k]x_i[k]. \quad (13)$$

Define  $\gamma_i[k] := (\tilde{T}_i[k]T_i[k])^{-1} / \sum_{j=1}^N (\tilde{T}_j[k]T_j[k])^{-1}$ , and  $\gamma[k] = [\gamma_1[k], \dots, \gamma_N[k]]^T$ . Then,  $\sum_{i=1}^N \gamma_i[k] = 1$ . We now have

$$x_i[k+1] = x_i[k](1 - \beta_i[k]E_i[k]) + \gamma_i[k] \sum_{j=1}^N x_j[k]\beta_j[k]E_j[k]. \quad (14)$$

In vector form, we have

$$\mathbf{x}[k+1] = A[k]\mathbf{x}[k], \quad (15)$$

where

$$\begin{aligned} A[k] &= A(\beta[k], \gamma[k], \mathbf{E}[k]) \\ &= \text{diag}(1 - \beta_1[k]E_1[k], \dots, 1 - \beta_N[k]E_N[k]) \\ &\quad + \gamma[k](\beta_1[k]E_1[k], \dots, \beta_N[k]E_N[k]). \end{aligned} \quad (16)$$

We see that  $\mathbf{x}[k]$  forms a discrete-time Markov Chain on the general space  $\Sigma$ . For any  $k$ ,  $A[k]$  is a non-negative random matrix and column stochastic matrix [7], [14], [15], and the shape of  $A[k]$  is determined by  $\mathbf{E}[k]$  ( $\beta[k]$  and  $\gamma[k]$  also influence  $A[k]$ , but they do not determine the shape of  $A[k]$ ). The modeling of  $\mathbf{E}[k]$  thus determines the properties of this Markov Chain, and different modelings of  $\mathbf{E}[k]$  lead to different fairness results of these general AIMD algorithms.

### C. Synchronized Backoff

All the prior work, no matter synchronized backoff or unsynchronized backoff modeling, simplifies the system by making the following assumption:

$$\beta_i[k] \equiv \beta, T_i[k] \equiv \tilde{T}_i[k] \equiv T_i, \gamma_i[k] \equiv \gamma_i = \frac{T_i^{-2}}{\sum_{j=1}^N T_j^{-2}}, \forall^5 i, k. \quad (17)$$

For the synchronized backoff model, it is assumed in [21] that

$$q_i[k] \equiv 1, \forall i, k, \text{ and hence } \mathbf{E}[k] \equiv (1, \dots, 1)^T. \quad (18)$$

Under this modeling, the system is deterministic and it is proved in [21] that  $\mathbf{x}[k]$  converges to an equilibrium point  $\mathbf{x}^*$  (accordingly,  $\mathbf{W}[k]$  converges to  $\mathbf{W}^*$ ) and

$$\bar{W}_i[k] := \sum_{h=1}^N W_i[h]/k \rightarrow W_i^* \text{ and } \bar{x}_i[k] := \sum_{h=1}^N x_i[h]/k \rightarrow x_i^*,$$

and

$$W_i^* \propto 1/T_i, \text{ and } x_i^* \propto 1/T_i^2. \quad (19)$$

Equation (19) can be easily derived by equating  $x_i[k+1] = x_i[k] = x_i^*$  in (14):

$$x_i^* = x_i^*(1 - \beta) + \gamma_i \sum_{j=1}^N \beta x_j^* \Rightarrow x_i^* \propto \gamma_i \propto 1/T_i^2$$

We notice that (19) is inconsistent with (1) for the standard TCP. The reason is that under the synchronized backoff modeling, different flows see the same number of loss events, and larger (respectively, smaller) rate flows see smaller (respectively, larger) loss event probabilities. The resource allocation in this synchronized backoff modeling is very unfair to long RTT flows, compared to the widely accepted belief that the standard TCP's resource allocation is  $x_i^* \propto 1/T_i$  if the window backoffs are unsynchronized. Simulation studies, Internet experiments and our analysis later show that synchronized backoff happens if  $N$  is small and congestion is very heavy (many packets are dropped).

### D. Unsynchronized backoff: Prior Modeling

If  $N$  is large, or if only a few packets are dropped at one congestion event, then not all flows see packet losses, and thus we need an unsynchronized backoff model, where  $q_i[k] < 1$ , and  $\mathbf{E}[k]$  can pick any value in  $\mathbb{E}$ . All the prior work, like [3]–[5], [29], [33] assumes that (17) holds and assumes that the the sequence  $\{\mathbf{E}[k]\}_{k \in \mathbb{N}}$  is identical independent distributed (i.i.d.):

$$\rho_l[k] \equiv \rho_l, \forall k, \forall l, q_i[k] \equiv q_i > 0, \forall i, k. \quad (20)$$

Under this assumption, it is shown in [33] and [29] that there exists a unique invariant distribution  $\pi$  of  $\mathbf{x}[k]$ , starting from any initial state, the distribution of  $\mathbf{x}[k]$  converges to this invariant distribution, and

$$\bar{W}_i[k] \rightarrow E_{\pi}[W_i[k]] \propto 1/q_i T_i, \text{ and } \bar{x}_i[k] \rightarrow E_{\pi}[x_i[k]] \propto 1/q_i T_i^2. \quad (21)$$

<sup>5</sup>Henceforth,  $\forall i$  stands for all users  $i$ ,  $\forall k$  stands for all congestion events  $k$ , and  $\forall l$  stands for all integer  $l$  between 1 and  $\mu$ .

Equation (21) can be easily derived by taking conditional expectations of  $x_i[k+1]$  given  $\mathbf{x}[k]$  in (14):

$$E[x_i[k+1]|\mathbf{x}[k]] = x_i[k](1 - \beta q_i) + \gamma_i \sum_{j=1}^N x_j[k] \beta q_j$$

Taking expectation over  $\mathbf{x}[k]$ , we have

$$E[x_i[k+1]] = E[x_i[k]](1 - \beta q_i) + \gamma_i \sum_{j=1}^N E[x_j[k]] \beta q_j$$

Equating  $E[x_i[k+1]]$  and  $E[x_i[k]]$  under the invariant distribution  $\pi$ , we get  $E[x_i[k]] \propto \gamma_i/q_i \propto 1/q_i T_i^2$ . From (21), for fixed  $q_i$ 's and varying  $T_i$ 's, still we get (19) and the resource allocation is the same as the unsynchronized modeling. Intuitively, if  $q_i[k]$  does not depend on  $W_i[k]$  or  $x_i[k]$ , we cannot guarantee that larger (respectively, smaller) throughput users see more (respectively, fewer) loss events, and thus the loss event probability seen by different RTT users are still different. This is just the same phenomenon that occurs in a synchronized model, and it is not surprising that the synchronized backoff model and unsynchronized backoff model lead to the same result in (19).

However, when the window backoff is not synchronized, AIMD is widely observed to have the following fairness property:

$$p_i \approx p_j, W_i \approx W_j, \forall i, j, \text{ and } \bar{x}_i \propto 1/T_i, \quad (22)$$

and  $p_i$  and  $\bar{W}_i$  are related as in (1). Equation (21) is inconsistent with this widely observed result, unless we assume that  $q_i \propto 1/T_i$ . However, that assumption is too arbitrary and not rooted, as it is not realistic to relate the window backoff probability with the round trip time. We now first study the most general unsynchronized window backoff modeling, and then make a realistic assumption to derive a fairness property which is consistent to (22).

#### E. Unsynchronized backoff: New General Modeling

In all the prior work,  $T_i[k]$ ,  $\tilde{T}_i[k]$  and  $\beta_i[k]$  are assumed to be constants. We first make the following assumption to allow more generality in our new modeling.

**Assumption 1.** For each user  $i$  and each congestion event  $k$ ,  $T_i[k]$ ,  $\tilde{T}_i[k]$ , and  $\beta_i[k]$  are random variables which take values in finite sets. All the sequences  $\{T_i[k]\}_{k \in \mathbb{N}}$ ,  $\{\tilde{T}_i[k]\}_{k \in \mathbb{N}}$ ,  $\{\beta_i[k]\}_{k \in \mathbb{N}}$  are i.i.d. sequences. All these random variables ( $T_i[k]$ ,  $\tilde{T}_i[k]$ ,  $\beta_i[k]$ ) for different users are independent to each other.  $\beta_i[k]$ ,  $\forall i$  is independent to  $T_j[k]$ ,  $\tilde{T}_j[k]$ ,  $\forall j$ .

From this assumption, the sequence of  $\{\gamma_i[k]\}_{k \in \mathbb{N}}$  is also i.i.d, and  $\gamma[k]$ 's are also independent to  $\mathbf{x}[k]$ 's and  $\beta[k]$ 's. For these random variables, we define their expected values:  $\hat{T}_i = E[T_i[k]]$ ,  $\hat{T}'_i = E[\tilde{T}_i[k]]$ ,  $\hat{\gamma}_i = E[\gamma_i[k]]$ , and  $\hat{\beta}_i = E[\beta_i[k]]$ . From the definition of  $\gamma_i[k]$ , we have

$$\hat{\gamma}_i \approx (\hat{T}_i \hat{T}'_i)^{-1} / \sum_{i=1}^N (\hat{T}_i \hat{T}'_i)^{-1}. \quad (23)$$

We then consider the assumption on the loss event random variable. The contradictive results derived in the prior unsynchronized backoff model is because of the assumption that

$\{\mathbf{E}[k]\}_{k \in \mathbb{N}}$  is an i.i.d. sequence, and  $\rho_i[k]$  and  $q_i[k]$  are constant independent of  $k$  or  $\mathbf{x}[k]$ . This assumption is unrealistic, since intuitive a flow will back off with different probability as the rate  $x_i[k]$  varies. From this intuition, we replace the assumption in (20) with the following assumption:

**Assumption 2.** The distribution of  $\mathbf{E}[k]$  depends on  $\mathbf{x}[k]$  and  $\{\mathbf{E}[k]\}_{k \in \mathbb{N}}$  are not i.i.d. Furthermore,  $\rho_i[k]$  and  $q_i[k]$  are continuous functions of  $\mathbf{x}[k]$ .

From this assumption, we can write

$$\rho_i[k] = \rho_i(\mathbf{x}[k]), \forall i, k, \text{ and } q_i[k] = q_i(\mathbf{x}[k]), \forall i, k, \quad (24)$$

where  $\rho_i(\cdot)$  and  $q_i(\cdot)$  are continuous functions. From this new general window backoff modeling, we can prove the following theorems.

**Theorem IV.1.** Under Assumptions 1 and 2, the Markov Chain defined in (15) and (16) has a unique invariant distribution, and starting from any initial state, the distribution of  $\mathbf{x}[k]$  converges to this invariant distribution. Moreover, the ergodicity of the Markov Chain is satisfied, i.e., for any continuous function  $h(\cdot) : \Sigma \rightarrow \mathbb{R}$ , the time average of  $h(\mathbf{x}[k])$  equals the expected value of  $h(\mathbf{x}[k])$  under the invariant distribution.

*Proof.* See [23].  $\square$

**Theorem IV.2.** Under the unique invariant distribution of the Markov Chain defined in (15) and (16), the following equation holds:

$$\frac{\hat{\beta}_i}{\hat{\gamma}_i} E[x_i[k] q_i(\mathbf{x}[k])] = C_1, \forall i, \quad (25)$$

and

$$\xi_i \hat{\beta}_i E[W_i[k] T_i q_i(\mathbf{x}[k])] \approx C_2, \forall i, \quad (26)$$

where  $\xi_i := \hat{T}'_i / \hat{T}_i$ , and  $C_1$  and  $C_2$  are constants independent of  $i$ .

*Proof.* Since  $\gamma[k]$  and  $\beta[k]$  are independent of  $\mathbf{x}[k]$ , their conditional expectations given  $\mathbf{x}[k]$  are the same as their expected values. Taking expectation of  $x_i[k+1]$  given  $\mathbf{x}[k]$  in (14), we have

$$E[x_i[k+1]|\mathbf{x}[k]] = x_i[k] - \hat{\beta}_i q_i(\mathbf{x}[k]) x_i[k] + \hat{\gamma}_i \sum_{j=1}^N \hat{\beta}_j q_j(\mathbf{x}[k]) x_j[k]. \quad (27)$$

Under the invariant distribution,

$$\begin{aligned} E[x_i[k]] &= E[x_i[k+1]] = E[E[x_i[k+1]|\mathbf{x}[k]]] \\ &= E[x_i[k]] - \hat{\beta}_i E[q_i(\mathbf{x}[k]) x_i[k]] \\ &\quad + \hat{\gamma}_i \sum_{j=1}^N \hat{\beta}_j E[q_j(\mathbf{x}[k]) x_j[k]]. \end{aligned} \quad (28)$$

So we have  $E[x_i[k] q_i(\mathbf{x}[k])] \hat{\beta}_i / \hat{\gamma}_i$  is independent of  $i$  and we have proved (25). Plugging in (23), we get (26).  $\square$

### F. Specific Modeling on $\rho_l(\cdot)$ and $q_i(\cdot)$

From Theorem IV.2, we see that the resource allocation depends on the form of  $q_i(\cdot)$ . If  $q_i(\cdot)$  is constant, it is exactly the same as the prior results in [33]. If  $q_i(\mathbf{x}[k]) \propto x_i[k]$ , then we can get that  $\xi_i \hat{\beta}_i E[(W_i[k])^2]$  is the same for all users. For both the standard TCP and TCP-Illinois,  $\hat{\beta}_i$ 's are the same for all users. From the definition of  $\xi_i$ , we know that  $T_i^P/\hat{T}_i \leq \xi_i \leq 1$  and  $\xi_i \approx \xi_j, \forall i, j$  in general. Then, we get  $E[(W_i[k])^2]$  is approximately the same for every user  $i$ . So if we can derive that  $q_i(\mathbf{x}[k]) \propto x_i[k]$  approximately, then we can derive the widely observed fairness property of AIMD that all window sizes are in the same level for different RTT users. Toward that end and to make the modeling realistic, we make the following assumption:

**Assumption 3.** *At each congestion event, the total number of packets dropped is a random variable, and its distribution is independent of  $k$ . Furthermore, for any packet dropped at congestion event  $k$ , the probability that it belongs to flow  $i$  is  $x_i[k]/C'$ .*

This assumption is justified by the following reasoning: since the total arrival rate is independent of  $k$ , so is the distribution for the total number of packets dropped; since the probability of an arbitrary packet belonging to flow  $i$  is  $x_i[k]/C'$ , so is the probability of a dropped packet belonging to flow  $i$ . From this assumption, we denote by  $M$  the random variable indicating the total number of packets dropped in one congestion event<sup>6</sup>, let  $p_D(m) = \text{Prob}(M = m)$  for any positive integer  $m$  ( $m \geq 1$  since at least one packet is dropped at each congestion event), and let  $\hat{M} = E[M]$ . Then, we have

$$\begin{aligned} q_i(\mathbf{x}[k]) &= 1 - \text{Prob}(\text{no dropped packets from flow } i) \\ &= \sum_{m=1}^{\infty} p_D(m) [1 - (1 - \frac{x_i[k]}{C'})^m] = f(x_i[k]), \end{aligned} \quad (29)$$

where  $f(x) := \sum_{m=1}^{\infty} p_D(m) [1 - (1 - \frac{x}{C'})^m]$  is a strictly increasing continuous function in  $x \in [0, C']$ . We then study  $\rho_l(\cdot)$ . For a specific  $l \in \{1, 2, \dots, \mu\}$ , suppose  $E_i^{(l)} = 1$  if  $i \in \{i_1, i_2, \dots, i_H\} \subset \{1, 2, \dots, N\}$ , where  $H \leq N$ , and  $E_i^{(l)} = 0$  otherwise. Then,  $\text{Prob}(E[k] = E^{(l)} | M = m) = 0$  if  $m \leq H$ . If  $m \geq H$ , we have

$$\begin{aligned} \rho_{l,m}(\mathbf{x}[k]) &:= \text{Prob}(E[k] = E^{(l)} | M = m) \\ &= \sum_{m_1, \dots, m_H} (\frac{x_{i_1}[k]}{C'})^{m_1} (\frac{x_{i_2}[k]}{C'})^{m_2} \dots (\frac{x_{i_H}[k]}{C'})^{m_H} \binom{m}{m_1, m_2, \dots, m_H} \end{aligned}$$

where the  $\sum$  is over all  $m_h \geq 1, \forall h \in \{1, 2, \dots, H\}$ , and  $\sum_{h=1}^H m_h = m$ . And we have

$$\rho_l(\mathbf{x}[k]) = \text{Prob}(E[k] = E^{(l)}) = \sum_{m=H}^{\infty} p_D(m) \rho_{l,m}(\mathbf{x}[k]). \quad (30)$$

So we have given the formula of  $\rho_l(\mathbf{x})$  and shown that  $\rho_l(\mathbf{x})$  is a continuous function of  $\mathbf{x}$ .

<sup>6</sup>When  $M$  is much larger than  $N$ , we call it a heavy congestion. When  $M$  is much smaller than  $N$ , we call it a light congestion.

### G. General Discussions on the Fairness Property

As shown in Theorem IV.2, the fairness property depends on the form of  $f(\cdot)$ . The exact form of  $f(\cdot)$  depends on the distribution of  $M$ , and is thus unknown if  $p_D(\cdot)$  is unknown. However, we can bound  $f(x)$  in the general case and approximate  $f(x)$  for some special cases. Since

$$1 - \frac{x}{C'} \geq (1 - \frac{x}{C'})^m \geq 1 - \frac{mx}{C'}, \quad \forall 0 \leq x \leq C',$$

we have

$$\begin{aligned} \frac{x}{C'} &= \sum_{m=1}^{\infty} p_D(m) [1 - (1 - \frac{x_i[k]}{C'})^m] \\ &\leq f(x) \leq \sum_{m=1}^{\infty} p_D(m) [1 - (1 - \frac{mx_i[k]}{C'})] = \frac{\hat{M}x}{C'} \end{aligned} \quad (31)$$

and when  $C'/x \gg \hat{M}$  (which holds if  $N \gg \hat{M}$ ),

$$f(x) \approx \sum_{m=1}^{\infty} [1 - (1 - \frac{mx}{C'})] p_D(m) = \frac{\hat{M}x}{C'} \quad (32)$$

From (29) to (32),  $q_i[k] \rightarrow 1$  as  $\hat{M} \gg N$ , and  $q_i[k] \rightarrow \frac{\hat{M}x}{C'}$  as  $\hat{M} \ll N$ . So heavy congestion ( $\hat{M} \gg N$ ) leads to synchronization, and light congestion ( $\hat{M} \ll N$ ) leads to the proportionality of  $q_i[k]$  on  $x_i[k]$ .

For the light congestion case,  $f(x_i[k]) \propto x_i[k]$ , and approximately we have

$$E[(W_i[k])^2] = E[(W_j[k])^2], \quad \forall i, j \in \{1, 2, \dots, N\}. \quad (33)$$

If the  $N$  users have the same RTT, the invariant distribution of the Markov Chain is unchanged if we swap two users. Thus,  $E_{\pi}[W_i[k]] = E_{\pi}[W_j[k]], \forall i, j$ , and we get that all the users share the bandwidth equally, which is intuitively true. If two users have different RTTs, since  $x_1[k] + x_2[k] = C'$ , we have  $\text{Var}(x_1[k]) = \text{Var}(x_2[k])$ , and thus from (33), we have  $E[W_1[k]] = E[W_2[k]]$ .

If  $N > 2$  users have different RTTs, the equality of  $E[W_i[k]]$  for all  $i$  is not guaranteed. However, if either of the two following conditions hold: (i) users with larger  $E[W_i]$  also have larger  $\text{Var}(W_i)$ ; (ii)  $\text{Var}(W_i) \ll E^2[W_i]$ , then, we have exactly or approximately  $E[W_i]$  is the same for all users  $i$ . From the ergodicity of the Markov Chain,

$$\bar{W}_i \approx \bar{W}_j, \quad \forall i, j \text{ and } \bar{x}_i \propto 1/\hat{T}_i, \quad \forall i. \quad (34)$$

Note that the average of  $W$  and  $x$  is over their values at the congestion events, not over all time. Since a general AIMD algorithm can yield any window size curve, it is a challenging problem to compute the average  $W$  and  $x$  over all time, and it is an open problem whether the all time average  $W$  is the same for all users and the all time average  $x$  is inversely proportional to RTT. Our simulations support the claim that (34) hold for all time average also, which is the widely believed fairness condition for AIMD if the backoff is unsynchronized. Since our analysis works for both the standard TCP and TCP-Illinois, we know that the fairness property of TCP-Illinois is similar to the standard TCP.

We have performed a large number of simulations on the evolution of the Markov Chain defined in (15) and (16). We vary  $N$  from 3 to 10. For each  $N$ , we select three probability distribution of  $M$ : (i) light congestion,  $\hat{M} < N$ ; (ii)

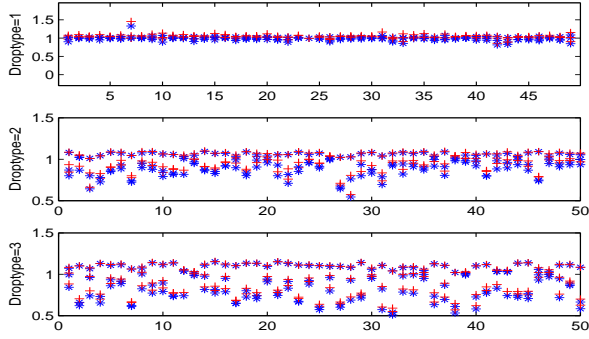


Fig. 2. The  $\bar{x}_i/x_i^*$  and  $\sqrt{x_i^2}/x_i^*$  ratios when  $N = 3$ .

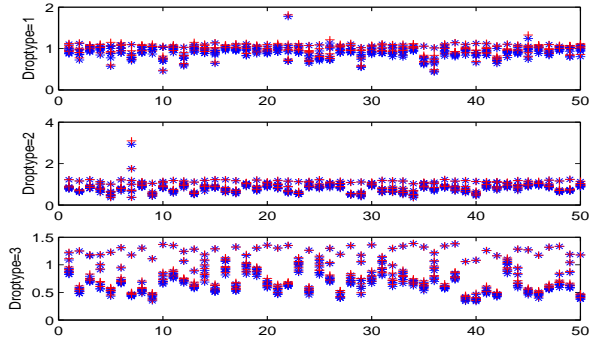


Fig. 3. The  $\bar{x}_i/x_i^*$  and  $\sqrt{x_i^2}/x_i^*$  ratios when  $N = 7$ .

middle congestion,  $\hat{M} \approx N$ ; (iii) heavy congestion,  $\hat{M} > N$ . For each scenario, we perform 50 simulations and in each simulation,  $\gamma[k]$ ,  $\beta[k] = \beta_i[k], \forall i$ ,  $\mathbf{x}[0]$ , and  $M$  are randomly generated,  $A[k]$  matrix is generated from Assumption 3, and the sample path of the Markov Chain is derived. For each sample path, we average 500 points after the distribution converges to compute  $\bar{x}_i$ ,  $\sqrt{x_i^2}, \forall i$ . We also compute  $x_i^*$  for all  $i$  by assuming that (34) holds. We plot  $\bar{x}_i/x_i^*$  and  $\sqrt{x_i^2}/x_i^*$  for all user  $i$  and all simulations performed, and the results are shown in Fig. 2 to Fig. 4<sup>7</sup>. From the figures, we have the following observations: (i)  $\bar{x}_i$  and  $\sqrt{x_i^2}$  are always very close to each other, which means that  $\text{Var}(x_i[k]) \ll (E[x_i[k]])^2$ . (ii) For the light congestion,  $\bar{x}_i/x_i^*$  is always close to 1, which means that (34) holds. As the congestion becomes heavier, the range of  $\bar{x}_i/x_i^*$  becomes wider and the difference between  $W_i$ 's becomes larger. These simulations support our analysis of the fairness property at light congestion and middle/heavy congestion.

**Remark 1.** Above we get (34) as the approximated fairness condition. For the exact form of the fairness, from (31), ergodicity of the Markov Chain, and the definition of  $\xi_i$ , we have

$$\frac{T_j^P}{\hat{M}\hat{T}_j} \leq \frac{\bar{W}_i^2}{\bar{W}_j^2} \leq \frac{\hat{M}\hat{T}_i}{T_i^P}, \quad \forall i \neq j. \quad (35)$$

<sup>7</sup>Due to limit of space, we only provide the case for  $N = 3, 7, 10$ .

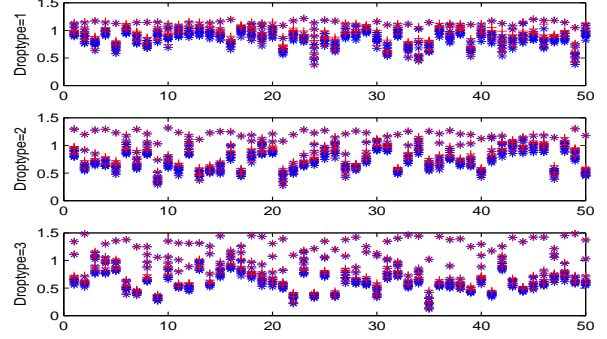


Fig. 4. The  $\bar{x}_i/x_i^*$  and  $\sqrt{x_i^2}/x_i^*$  ratios when  $N = 10$ .

So the smaller  $\hat{M}$  is, the tighter the bound is and the closer  $\bar{W}_i^2$  and  $\bar{W}_j^2$  are. In extreme cases, as  $\hat{M} \rightarrow \infty$  (heavy congestion and synchronization happens), (35) is meaningless and  $\bar{W}_i$  and  $\bar{W}_j$  can be significantly different; while as  $\hat{M} \rightarrow 1$  (small congestion and no synchronization),  $\bar{W}_i^2 \approx \bar{W}_j^2$ . If the variance is much smaller than the square of the expected value, then  $\bar{W}_i \approx \bar{W}_j$  also.

**Remark 2.** Usually when we say “synchronized”, we mean that the number of loss events  $\lambda$  ( $\lambda \propto f(x)$ ) is the same for all users, and as a result,  $W \propto 1/T$  and  $x \propto 1/T^2$ . When we say “unsynchronized”, we mean that  $p$  ( $p \propto f(x)/x$ ) is the same for all users, and as a result,  $W$  is independent of  $T$  and  $x \propto 1/T$ . As we have shown, the backoff is synchronized if  $\hat{M} \gg N$ , and unsynchronized if  $\hat{M} \approx 1$  or  $N \gg \hat{M}$ . However, there are many cases between these two extreme cases: for the general  $\hat{M}$  and  $N$  relationship, a larger rate user has a larger  $\lambda$  ( $f(x)$  is increasing in  $x$ ), but a smaller  $p$  ( $f(x)/x$  is decreasing in  $x$ ), and we call these scenarios “partial-unsynchronized”, and (35) gives the bound of the allocation for the general partial-unsynchronized case. If we write  $x \propto 1/T^v$ , then  $v = 2$  for completely synchronized case and  $v = 1$  for completely unsynchronized case and  $1 \leq v \leq 2$  for general partial-unsynchronized case.

**Remark 3.** The analysis holds for both the standard TCP and TCP-Illinois, since they both have the following feature: all users see the same  $\alpha(t), \forall t$ . However, the analysis does not apply to some other general AIMD algorithms, such as HS-TCP, Scalable-TCP and H-TCP, since for those algorithms, different users see different  $\alpha(t)$  and  $\beta[k]$  values, and  $\alpha(t)$  and  $\beta[k]$  may be dependent of  $W$ .

We now see the factors that influence  $\hat{M}$ . Consider the homogeneous case and suppose the system is slotted with each slot being one RTT. Since the pipe can hold at most  $CT + B$  packets and  $W_i$  increases by  $\alpha_i$  within each slot,  $\sum_{i=1}^N W_i \in [CT + B + 1 - \sum_{i=1}^N \alpha_i, CT + B]$  in the slot just before congestion, and  $\sum_{i=1}^N W_i \in [CT + B + 1, CT + B + \sum_{i=1}^N \alpha_i]$  in the slot of congestion. As a result, anywhere from 1 to  $\sum_{i=1}^N \alpha_i$  packets could be dropped at one congestion event, and we know that the congestion is heavier if the increment before congestion is larger. This explains the be-

havior (as demonstrated in Section VI) that the convex curve algorithms, like HS-TCP, yields heavy congestion regularly, while the concave curve algorithms, like TCP-Illinois, does not.

#### H. Compatibility with The Standard TCP

If TCP-Illinois coexists with the standard TCP, we can show that (see [23] for the proof), all Reno users share the same average window size  $\bar{W}_R$  and all Illinois users share the same average window size  $\bar{W}_{IL}$ , and

$$\frac{\bar{W}_{IL}}{\bar{W}_R} \approx \sqrt{\frac{\alpha_{IL}^*}{2\beta_{IL}^*}} \approx \sqrt{\frac{\alpha_{IL}^*}{2\beta_{max}}},$$

where  $\alpha_{IL}^*$  lies in between  $\alpha_{max}$  and  $\alpha_{min}$  and is usually slightly larger than 1, and  $\beta_{IL}^*$  is smaller than and approximately equals to  $\beta_{max}$ , which is 1/2 in the default setting. This means that in a network with both TCP-Illinois and TCP-Reno users, the TCP-Reno users will not suffer a significant degradation in performance. Furthermore, unlike TCP-Vegas which performs poorly when used with TCP-Reno, TCP-Illinois actually performs better than TCP-Reno, thus providing the right incentive for users to switch to TCP-Illinois.

#### V. PROPERTIES AND REQUIREMENT SATISFACTION

In Section II, we listed some requirements for the new TCP variant to satisfy, and in Section IV, we showed that TCP-Illinois maintains the intra protocol fairness the same way as the standard TCP, satisfies the stability and scalability requirement, avoids heavy congestion, and is compatible with the current TCP. In this section, we consider the remaining requirements.

In Section II, we listed some requirements for the new TCP variant to satisfy, and in Section IV, we showed that TCP-Illinois maintains the intra protocol fairness the same way as the standard TCP, satisfies the stability and scalability requirement, avoids heavy congestion, and is compatible with the current TCP. In this section, we consider the remaining requirements.

Since  $\alpha$  decreases with increasing  $W$ , the  $W$  curve is concave. We can show that the curve is actually first linear, and then a parabola. The proof is omitted due to space limitations and is available in [23]. In [23], we also show that TCP-Illinois achieves a better average throughput than the standard TCP for any route buffer size  $B$ , and its average throughput increases as  $B$  increases, since compared with the standard TCP, TCP-Illinois increases its rate to full utilization faster and stays around full utilization longer (the length of time it stays around full utilization increases with increasing  $B$ ). Thus the requirements of efficiency, router buffer independence, and incentive to switch are all met.

From Section IV, we see that convergence in  $k$  for TCP-Illinois is the same as that for the standard TCP. So the response time is only determined by the time interval between two consecutive congestion events. We can show that if  $\alpha_{min} \geq d_{max}/T_{max}$ , then this time interval of TCP-Illinois

is similar to that of the standard TCP (see [23] for a proof), and thus the responsiveness requirement is also satisfied.

In lossy networks such as wireless networks, many packets are dropped not due to congestion. These packet drops greatly reduce the throughput for the standard TCP, but for TCP-Illinois, the degradation is not as severe, since when a packet is dropped before congestion, the average queuing delay is always almost zero, and thus  $\beta \approx \beta_{min}$  and  $\alpha \approx \alpha_{max}$  always, and TCP-Illinois is essentially an AIMD algorithm with a larger  $\alpha = \alpha_{max}$  and smaller  $\beta_{min}$ . Since  $W \propto \sqrt{\alpha/\beta}p$ , the ratio of the average window size of TCP-Illinois over that of the standard TCP can be up to  $\sqrt{\alpha_{max}/(2\beta_{min})}$ . This improvement is significant. For example, if  $\alpha_{max} = 9, \beta_{min} = 1/8$ , then  $W_{Illinois}$  can be up to  $6W_{Reno}$ .

#### VI. SIMULATION RESULTS

In this section, we provide simulation results to validate the properties of TCP-Illinois and compare its performance with TCP-Reno and HS-TCP. Throughout, one bottleneck link is shared by one or multiple users, which may choose TCP-Reno, HS-TCP, TCP-Illinois, or TCP-Vegas. For HS-TCP, all the default parameter settings are used. For TCP-Vegas,  $W$  increases if  $diff < \gamma$  and decreases if  $diff > \gamma$ . For TCP-Illinois, without explicit explanation, we set  $\alpha_{max} = 10, \alpha_{min} = 0.1, \beta_{max} = 1/2, \beta_{min} = 1/8, W_{thresh} = 10, \eta_1 = 0.0, \eta_2 = 0.1$ , and  $\eta_3 = 0.8$ .

##### A. Single User: Efficiency Property

We first perform simulations for a single user scenario, with  $C = 100$  Mbps,  $B = 100$  packets<sup>8</sup>, and  $T_p = 100$  ms. The window sizes are plotted in Fig. 5. The simulations clearly demonstrate the concave nature of the curve of TCP-Illinois and show that TCP-Illinois achieves a larger average window size than TCP-Reno. For HS-TCP, we have chosen the Reno base, NewReno base and SACK base, and we have found that HS-TCP generates timeouts frequently for Reno and NewReno bases, and only works well if SACK is used. This supports our claim that HS-TCP causes heavy congestion. Numerically, the average send rates for TCP-Reno, SACK based HS-TCP<sup>9</sup>, and TCP-Illinois are 78.032, 87.324 and 91.304 Mbps, respectively. As a comparison to HS-TCP, if TCP-Illinois sets  $\beta_{max} = \beta_{min} = 0.125$  ( $\beta \approx 0.125$  for HS-TCP in this capacity range), then the average throughput is 95.727 Mbps.

##### B. Multiple Users: Fairness Property

We now perform simulations for multiple ( $N = 4$ ) users, which may choose Reno, HS-TCP or Illinois. We demonstrate the inter-protocol fairness (compatibility to Reno) of Illinois and HS-TCP in the homogeneous RTT scenario, where  $C = 100$  Mbps,  $B = 100$  packets, and  $T_p = 100$  ms; and demonstrate the intra-protocol fairness of Reno, Illinois and HS-TCP in the heterogeneous RTT scenario, where  $C$  and  $B$  are unchanged, but the  $RTT$ s for the four flows are

<sup>8</sup>The packet size is 1000 bytes throughout.

<sup>9</sup>Henceforth, we mean SACK based HS-TCP when we mention HS-TCP without specifying its base.

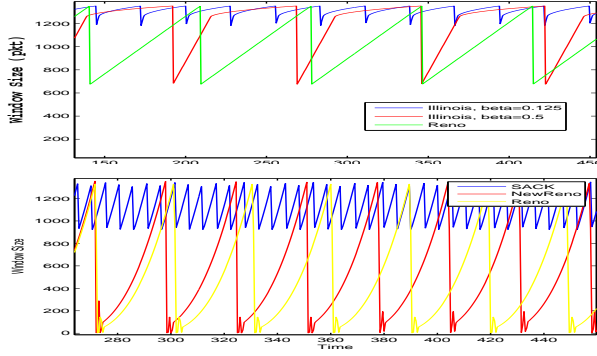


Fig. 5. Single user, TCP-Reno, HS-TCP, and TCP-Illinois. Top plot: Reno and Illinois (beta=0.125 and beta=0.5). Bottom plot: HS-TCP, with Reno, NewReno, and SACK bases.

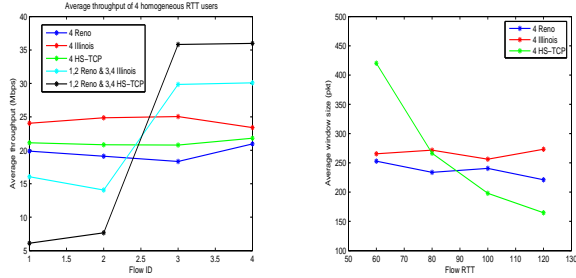


Fig. 6. Left: Average throughput of 4 homogenous RTT users. Right: Average window of 4 heterogeneous RTT users.

60, 80, 100, and 120 ms, respectively. The average throughput in the homogeneous scenario and average window size in the heterogeneous scenario are plotted in Fig. 6. From this figure, we see clearly that Illinois is more fair to competing Reno user and large RTT users than HS-TCP.

### C. Performance in Lossy/Wireless Networks

We then perform simulations for lossy/wireless links. It is a single link single user scenario, with the user choosing either TCP-Reno or TCP-Illinois, and the link randomly drops packets with dropping probability  $p_d$  much larger than the congestion loss probability (since  $p_d$  is large, the link is under utilized and there is no congestion loss at all in many cases). The capacity and buffer length of the link are 40 Mbps and 200 packets, respectively, and the propagation delay for the single user is 100 ms. Instead of choosing the default setting, the Illinois user sets  $\eta_1 = 0.2$ . We vary  $p_d$  values from 0.0005 to 0.05, and plot the average window size for Illinois and Reno and the ratio of these two multiplied by 20, as in the left plot of Fig. 7. From the plot, we see that  $W_{Illinois} \approx 4W_{Reno}$  in most cases. From (??), the ratio should be  $\sqrt{\alpha_{max}/(2\beta_{min})} = \sqrt{40} \approx 6.32$ . The difference of the ratio between simulation and analysis can be explained if we observe the window curve plot of Illinois and Reno, as in the right plot of Fig. 7. From the window curve plot, we see that timeout happens frequently for Illinois user, since the increment amount  $\alpha$  is very large before a packet loss happens. Equation (??) only considers the congestion avoidance phase, and timeout is the reason

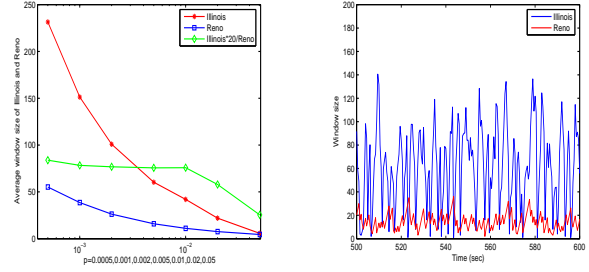


Fig. 7. Window sizes of Illinois Vs Reno in Lossy Networks. Left: Average window Vs  $p$ . Right: Window curve over  $t$  for  $p = 0.005$ .

that  $W_{Illinois}/W_{Reno}$  is around 4 instead of 6. Though, Illinois achieves a much better throughput than Reno in wireless networks.

### D. Performance with Noisy RTT Measurement

We finally perform simulations to compare the performance of TCP-Illinois and TCP-Vegas when the delay measurement is inaccurate. We consider a single link and two user scenario. For the link,  $C = 10$  Mbps and  $B = 50$  packets (correspondingly, the maximum queueing delay  $d_m = 40$  ms). For the users, either both choose TCP-Illinois or both choose TCP-Vegas, and the propagation delay is  $T_p = 60$  ms for each user. We now suppose that there is an extra white noise term in the RTT measurement, denoted by  $n$ , and let  $n$  be uniformly distributed between  $[0, 2\sigma]$  (the noise term is an extra delay due to reasons other than propagation and queueing, so it is nonnegative). Then,  $RTT = T_p + d + n$ , where  $d$  is the queueing delay. We vary the value of  $\sigma$  to vary the noise level and study the performance of TCP-Vegas and TCP-Illinois under noisy RTT measurement. For each protocol, we have two groups of simulations. In group one, both users face the noise term in the RTT measurement; and in group two, only one user faces the noise term and the other user measures RTT accurately. The average throughput of the users under different noise levels are plotted in Fig. 8. From Fig. 8, we see that as the noise level increases, TCP-Illinois is very robust to the noise, while TCP-Vegas is not: there is a threshold of  $\sigma$ , which depends on the  $\gamma$  parameter. If the noise level exceeds this threshold, the performance is degraded significantly: if both users have noise terms, both get a much smaller throughput than the noise free case and the link is under utilized; if one user has the noise term and the other does not, then the resource allocation is very unfair to the user with inaccurate RTT information. It is easy to explain the degradation when  $\sigma$  is larger than the threshold. At equilibrium, Vegas user satisfies

$$diff = \left( \frac{W}{T_p} - \frac{W}{T_p + d_a} \right) T_p = W \frac{d_a}{T_p + d_a} = \gamma \Rightarrow W^* = \gamma \frac{d_a^* + T_p}{d_a^*}, \quad (36)$$

where  $d_a^*$  and  $W^*$  are the equilibrium values of  $d_a$  and  $W$ . So  $W^*$  is an decreasing function of  $d_a^*$ , and  $d_a^*$  is a positive value such that the sum of  $W^*/(T_p + d_a^*)$  over all users equals the capacity  $C$ . If  $RTT = T_p + d_a + n$ , then since  $n$  may hit zero, still we have  $BaseRTT = T_p$ , and (36) becomes to the

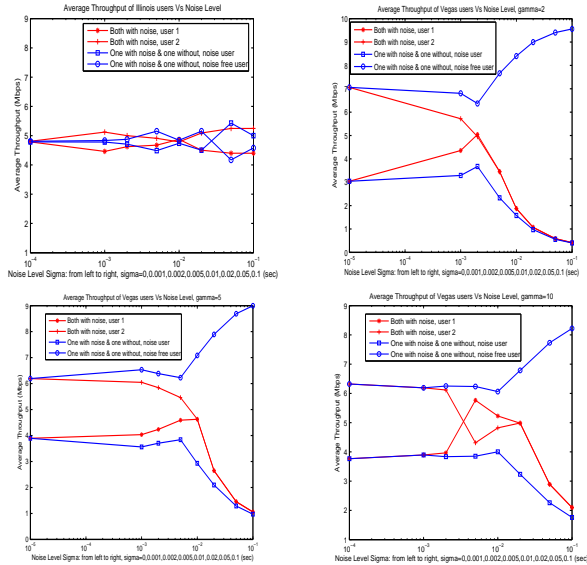


Fig. 8. Average Throughput Vs Noise Level. Top left: Illinois. Top right: Vegas,  $\gamma = 2$ . Bottom left: Vegas,  $\gamma = 5$ . Bottom right: Vegas,  $\gamma = 10$ .

following equation:

$$\begin{aligned} diff &= \left( \frac{W}{T_p} - \frac{W}{T_p + \bar{d}_a + \bar{n}} \right) T_p = W \frac{\bar{d}_a + \bar{n}}{T_p + \bar{d}_a + \bar{n}} = \gamma \\ \Rightarrow W &= \gamma \frac{\bar{d}_a + T_p + \bar{n}}{\bar{d}_a + \bar{n}} \text{ and } \bar{W} \approx \gamma \frac{\bar{d}_a + T_p + \bar{n}}{\bar{d}_a + \bar{n}}, \end{aligned} \quad (37)$$

where  $\bar{n}$ ,  $\bar{d}_a$  and  $\bar{W}$  are the time average values of  $n$ ,  $d_a$  and  $W$ . We see that if  $\bar{n} = \sigma \leq d_a^*$ , we can pick  $\bar{d}_a = d_a^* - \sigma$  so that  $\bar{W} = W^*$ ; and if  $\bar{n} = \sigma > d_a^*$ , then  $\bar{W}$  is definitely smaller than  $W^*$ . And when  $\sigma > d_a^*$ , as  $\sigma$  increases,  $\bar{W}$  decreases. So there exists a threshold of  $\bar{n} = \sigma$  such that if the noise level is smaller than this threshold, the performance can be similar to the noise free case; and if the noise level is larger than this threshold, the performance is degraded and the degradation becomes more significant as  $\sigma$  increases. Since this threshold approximately equals to  $d_a^*$  and  $d_a^*$  is proportional to  $\gamma$ , we know that this threshold is also proportional to  $\gamma$ , as shown in Fig. 8.

## VII. CONCLUSION

In this paper, we have considered some natural requirements for a new TCP protocol for high speed networks and have introduced a class of LDCA-CAIMD algorithms, which combines both loss and delay information to control the congestion. Specifically, the standard TCP's AIMD algorithm is modified such that the increment (respectively, decrement) amount for each ACK (respectively, loss) is a decreasing (respectively, increasing) function of the average queueing delay. Using this idea, a specific protocol called TCP-Illinois is designed. TCP-Illinois achieves a concave window size curve and a better throughput than the standard TCP, and maintains the fairness of the standard TCP. Various properties of TCP-Illinois are studied, and TCP-Illinois is shown to satisfy all the requirements for an ideal high speed TCP variant.

To analyze the fairness property of TCP-Illinois, a new stochastic matrix modeling of general AIMD algorithms is

introduced. Using this new model, we approximately derived the fairness property of TCP-Illinois and the standard TCP under different synchronization level. There are some open problems of the new model, which include: (i) the rigorous relationship between  $E[W_i]$  and  $E[W_j]$ ,  $\forall i \neq j$  for the  $N > 2$  heterogeneous users scenario; (ii), the relationship between  $E[\bar{W}_i]$  and  $E[\bar{W}_j]$ ,  $\forall i \neq j$ , where the average is over all time interval.

**Acknowledgement:** We thank Dr. F. Baccelli for helpful discussions.

## REFERENCES

- [1] J. Ahn, P. Danzig, Z. Liu, and L. Yan. Experience with TCP vegas: Emulation and experiment. In *Proceedings of ACM SIGCOMM*, 1995.
- [2] S. P. M. and R. L. Tweedie. *Markov Chains and Stochastic Stability*. Springer-Verlag London Ltd., 1993.
- [3] F. Baccelli and D. Hong. Interaction of TCP flows as billiards. Technical Report, INRIA Rocquencourt, April 2002.
- [4] F. Baccelli and D. Hong. The AIMD model for TCP sessions sharing a common router. In *Proceedings of 39th Annual Allerton Conf. on Communication, Control and Computing*, October 2001.
- [5] F. Baccelli and D. Hong. AIMD, fairness and fractal scaling of TCP traffic. In *Proceedings of IEEE Infocom*, June 2002.
- [6] R. G. Bartle and D. R. Sherbert. *Introduction to Real Analysis*. John Wiley & Sons, Inc., New York, 1982.
- [7] A. Berman and R. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. SIAM, 1979.
- [8] S. Biaz and N. Vaidya. Is the round-trip time correlated with the number of packets in flight. In *Proceedings Internet Measurement Conference (IMC)*, October 2003.
- [9] L. Brakmo, S. O'Malley, and L. Peterson. TCP vegas: New techniques for congestion detection and avoidance. In *Proceedings of ACM SIGCOMM Symposium*, pages 24–35, August 1994.
- [10] L. Elsner, I. Koltracht, and M. Neumann. On the convergence of asynchronous paracontractions with applications to tomographic reconstruction from incomplete data. *Linear Algebra Appl.*, pages 65–82, 1990.
- [11] S. Floyd. Highspeed TCP for large congestion windows. Internet draft, draft-floyd-tcp-highspeed-01.txt, December 2003, Available at <http://www.icir.org/floyd/hstcp.html>.
- [12] S. Floyd and T. Henderson. The new reno modification to TCPs fast recovery algorithm. RFC 2582, 1999.
- [13] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, August 1993.
- [14] A. Graham. *Nonnegative Matrices and Applicable Topics in Linear Algebra*. Ellis Horwood Limited, Chichester, England, 1987.
- [15] R. Horn and C. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.
- [16] V. Jacobson. Congestion avoidance and control. *ACM Computer Communication Review*, 18:314–329, August 1988.
- [17] V. Jacobson. Berkeley TCP evolution from 4.3-tahoe to 4.3-reno. In *Proceedings of the Eighteenth Internet Engineering Task Force*, July 1990.
- [18] C. Jin, D. Wei, S. H. Low, G. Buhrmaster, J. Bunn, D. H. Choe, R. L. A. Cottrell, J. C. Doyle, W. Feng, O. Martin, H. Newman, F. Paganini, S. Ravot, and S. Singh. FAST TCP: From theory to experiments, April 2003.
- [19] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *Proceedings on ACM Sigcomm*, 2002.
- [20] T. Kelly. On engineering a stable and scalable TCP variant. Cambridge University Engineering Department Technical Report CUED/F-INFENG/TR.435, June 2002.
- [21] D. Leith and R. Shorten. Analysis and design of synchronised communication networks. *Automatica*, 41:725–730, 2005.
- [22] D. Leith, R. Shorten, and Y. Li. H-TCP: A framework for congestion control in high-speed and long-distance networks. HI Technical Report, August 2005. Available at <http://www.hamilton.ie/net/htcp/>.

- [23] S. Liu, T. Başar, and R. Srikant. TCP-illinois: A delay and loss-based congestion control algorithm for high-speed networks. Technical Report, UIUC, 2006. Available at [http://www.ews.uiuc.edu/~shaoliu/tcpillinois\\_full.pdf](http://www.ews.uiuc.edu/~shaoliu/tcpillinois_full.pdf).
- [24] J. Martin, A. A. Nilsson, and I. Rhee. Delay-based congestion avoidance for TCP. *IEEE/ACM Transactions on Networking*, pages 356–369, June 2003.
- [25] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgement options. RFC 2018, April 1996, available at <http://www.icir.org/floyd/sacks.html>.
- [26] J. Mo, R. J. La, V. Anantharam, and J. C. Walrand. Analysis and comparison of TCP reno and vegas. In *Proceedings of IEEE INFOCOM*, 1999.
- [27] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of ACM SIGCOMM*, 1998.
- [28] R. S. Prasad, M. Jain, and C. Dovrolis. On the effectiveness of delay-based congestion avoidance. In *Proceedings of Second International Workshop on Protocols for Fast Long-Distance Networks*, 2004.
- [29] R. Shorten, F. Wirth, and D. Leith. A positive systems model of TCP-like congestion control: Asymptotic results. Submitted to *IEEE/ACM Transactions on Networking*, 2005.
- [30] W. Stevens. *TCP/IP Illustrated, Vol.1 The Protocols*. Addison-Wesley, 1994.
- [31] K. Tan, J. Song, Q. Zhang, and M. Sridharan. A compound TCP approach for high-speed and long distance networks. In *Proceedings of IEEE Infocom*, April 2006.
- [32] A. Tang, J. Wang, S. Low, and M. Chiang. Equilibrium of heterogeneous congestion control protocols. In *Proceedings of IEEE Infocom*, Miami, FL, March 2005.
- [33] F. Wirth, R. Stanojevic, R. Shorten, and D. Leith. Stochastic equilibria of AIMD communication networks. Accepted by *SIAM J. Matrix Analysis and its Applications*, 2005. Available at [http://www.hamilton.ie/chris/SIMAX\\_july28.pdf](http://www.hamilton.ie/chris/SIMAX_july28.pdf).
- [34] L. Xu, K. Harfoush, and I. Rhee. Binary increase congestion control for fast long-distance networks. In *Proceedings of IEEE INFOCOM*, 2004.
- [35] A. Zanella, G. Procissi, M. Gerla, and M. Y. Sanadidi. TCP westwood: Analytic model and performance evaluation. In *Proceedings of IEEE Globecom*, 2001.