New Congestion Control mechanism for TCP with Packet Pair scheme

YOSHIFUMI NISHIDA,† OSAMU NAKAMURA† and JUN MURAI†

Congestion Control Scheme of TCP/IP protocol suite is established by Transmission Control Protocol (TCP). Using self clocking scheme, TCP enables to maintain optimum connection status for network path quickly unless giving an excessive load to the network. However, in wide area networks, there are some obstructive factors for self clocking scheme of TCP.

In this paper, we describe the obstructive factors for self clocking scheme. We propose new congestion control scheme using packet pair scheme and traffic shaping scheme. To combine these schemes with TCP, new TCP option and modification to TCP's congestion control algorithms are added. By using our scheme, TCP enables to maintain self clocking scheme smoothly. We also implemented this scheme on a network simulator for evaluation and resulted over 20% efficiency compared to a normal TCP in symmetric communication and over 40% efficiency in asymmetric communication.

1. Introduction

Congestion Control Scheme of TCP/IP protocol suite is established by Transmission Control Protocol (TCP). TCP is a flexible communication protocol enabling low speed communication links such as telephone lines to a high speed links such as FDDI or an ATM network.

This TCP's flexibility is established by 'self clocking' method which takes advantage of acknowledgement segment. Self clocking is designed based on the following concepts.

- When packets are transmitted continuously from a high-speed link entering into a slow link, the interval of the packets is expanded.
- The transmission interval of acknowledge segments is equal to the receiving interval of transmitted data segments from sender.
- Using acknowledgement segment as a trigger of transmitting segment enables to attain optimum transfer rate for communication path.

The arrival interval of the acknowledgement segments matches the interval of the sender's segments which was expanded through the bottleneck link. This interval is optimum for the bottleneck link and is also optimum for the whole communication path. Therefore TCP uses acknowledgement segment as a trigger of transmitting segment. The transmit interval will match the bandwidth of the bottleneck link and will become optimal for communication path.

The slow-start algorithm was devised to make communication status approach to self clocking status[1)4)]. Slow-start algorithm adds 'congestion window' parameter to TCP's sliding window scheme. TCP cannot inject more than 'congestion window' segments of unacknowledged data into the network. By controlling congestion window, TCP adapts transfer rate to communication path. Slow-start has two data transfer phases, slow-start and congestion avoidance. Two phases have different congestion window control algorithm.

TCP enters in slow-start phase when new communication is started or when TCP increases traffic again after the congestion period.

In the slow-start phase, TCP increases the value of congestion window by 1 segment for each ACK received. Therefore congestion window is doubled in every 1 round trip time. Doubled congestion window soon exceed the capacity of bottleneck link and it causes congestion and segment losses.

When TCP decides that congestion period comes to an end, TCP increase the congestion window again. If increased window size reaches half of the window size when congestion was detected, TCP moves to the congestion avoidance phase from slow-start phase.

In the congestion avoidance phase, TCP increases congestion windows at the graceful rate by 1 segment in every 1 round trip time.

The goal of slow-start algorithm is to establish TCP connection into to self-clocking condition as quickly as possible. In short, by using

† Graduate School of Media and Governance, Keio University

dition as quickly as possible. In short, by using slow-start algorithm, TCP can approach best window size roughly in slow-start phase and approach to more precisely in congestion avoidance phase.

These techniques works well in general situation. However, there are some obstructive factors for self clocking scheme of TCP. The next section explains obstructive factors for self clocking scheme.

2. Obstructive factors for self clocking scheme

2.1 Asymmetric link

In wide area networks, there are some asymmetric links which differs from the path for data packets (forward path) and the path for acknowledgement packets (reverse path).

The self-clocking scheme of TCP presume symmetric links. In asymmetric links, transmission delay will occur in reverse path. This transmission delay affects presumption of bottleneck bandwidth in forward path.

Vern Paxson analyzed about 40000 end-to-end Internet route measurements and he studied routing symmetry[3]. He found 49% of the mesurements observed on asymmetric link that visited at least one different city. He also analyzed prevalence of asymmetry and reported prevalence of asymmetry is increasing.

2.2 Route alternation

In[3], Vern also analyzed route alternation. In his study, 87% of entire route alternation occurred in over 6hours to days time scale. The rest of route alternation occurred over 10minutes time scale. The percentage of route alternation in seconds time scale is insignificant.

2.3 Delayed ack

Most current TCP implementations adopt delayed ack scheme that delays transmission of acknowledgement packets. Delayed ack scheme does not transmit an acknowledgement packets the instant it receives data. Instead, delayed ack scheme expected to have data going in the same direction with acknowledgement packets and to have more acknowledgement packets arriving. Using delayed ack scheme reduce number of packet transmission. Most implementations use a 200msec delayed ack timer. Therefore delayed ack scheme will delay an acknowledgement packets up to 200msec to see if there is new acknowledgement or data to send with acknowledgement packets.

Delayed packet transmission extends interval of acknowledgement packets. This extended interval affects self-clocking scheme. However, delayed ack scheme has following advantages, so to disabling delayed ack scheme will cause some problems.

(1) asymmetric links
In asymmetric links, there are some cases that bandwidth of reverse path is too small for bandwidth of forward path. In such case, the traffic amount of acknowledgement packets will exceed capacity of reverse path's bandwidth, although the traffic amount of data packets is too small for forward path's bandwidth.
Delayed ack scheme that reduce traffic amount of acknowledgement packets is effective in this situation.

(2) reducing sender's overhead
When acknowledgement packets arrives at sender, timer interrupt will occur from network interface. The frequency of timer interrupt depends on number of acknowledgement packets. Delayed ack scheme reduces number of acknowledgement packets, therefore reduces overhead of data sender.

3. New Congestion Control Scheme for TCP

Our main goal is to develop new congestion control scheme for wide area networks. In previous section, we summarize some obstructive factors for self clocking scheme. Route alternation can affect self-clocking, however possibility of route alternation in one TCP connection is rare. Therefore we focus on route asymmetry and delayed ack.

The core specification of TCP is devised in late 1980s and TCP is designed simply enough to implement on computers in those days. Considering recent computer architecture, it is reasonable to add new features to TCP that require more processing capacity. Accordingly, we developed following functions for new TCP.

3.1 Fine timer interruption

Current TCP implementations use 200msec and 500msec timer interrupt. Considering recent computer architecture capacity, it is reasonable to implement 10msec timer interrupt. This timer interrupt enables more precise RTT measurement and enables to establish following functions.

3.2 Rate-based transmission control scheme

Current TCP transmission control scheme depends on window size. TCP cannot inject more than window size of unacknowledged data into the network. Therefore TCP control data transmission is based on RTT. In long delay networks, RTT becomes large so TCP's RTT-based transmission control becomes coarsely. This coarse transmission control leads bursty packet transmission. Using fine timer interruption to control data transmission reduces bursty data transfer in long delay networks.

3.3 Packet pair scheme

Packet pair scheme is devised by S.Keshav[5]. Packet pair scheme transmits two packets with a short interval. Analyzing interval of two packets which is expanded through bottleneck link, packet pair scheme estimate bandwidth of bottleneck link.

To evaluate packet pair scheme, Bolot transmitted a stream of UDP echo packets and measured its round trip time delay[6]. Carter and Crovella developed a tool `'bprobe'` to estimate bandwidth of bottleneck link by using packet pair scheme. The `bprobe` transmits a stream of ICMP echo packets and measured its round trip time delay[7]. In these researches, packet pair scheme showed good result of estimating bottleneck link's bandwidth.

However, to combine packet pair with TCP contains some problems. First, packet pair requires two data packets transmission at a time. However, TCP can not transmit two packets when window does not open enough for size of two packets. Second, more accurate packet pair techniques requires measurement of packet pair's interval at receiver side. However, TCP has only RTT measurement scheme at sender side. RTT measurement is affected asymmetric links. Third, delayed ack algorithm may affects packet pair measurements. We discuss the solution of this problem in the next section.

4. Design of New TCP

To establish new congestion control scheme, we designed following functions.

4.1 Basic function

4.1.1 Rate-based transmission control function

Rate-based transmission control mechanism is established by 10msec timer interrupt routine. 10msec timer interrupt routine is invoked every 10msec. This routine maintains timer counter and has traffic shaping function using leaky buckets method.

4.1.2 PacketPair Option

To combine packet pair scheme with TCP, we developed new tcp option: packet pair option. Figure 1 shows the format of packet pair option. Packet pair option consist of following three types.

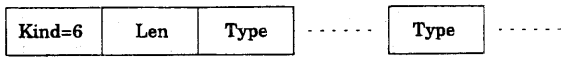

Fig. 1 PacketPair option format

(1) Packet Pair 1

The packet pair 1 type is to be used by data sender. The first packet of packet pair is transmitted with this option. Packet Pair 1 type option contains pair-id information to distinguish pair of packets at receiver side. When packet with packet pair 1 type option arrived at receiver side, receiver stores timer counter that maintains timer interrupt routine. The acknowledgement of packet with packet pair 1 type option can be delayed by delayed ack algorithm. Figure 2 shows example of packet pair 1 format.

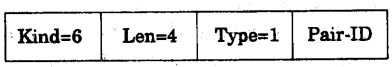

Fig. 2 PacketPair 1 format

(2) Packet Pair 2

The packet pair 2 type is to be used by data sender. The second packet of packet pair is transmitted with this option. The value of pair id information in packet pair 2 type option is same as pair of pair 1 type option's pair id.

When packet with packet pair 2 type option arrived at receiver side, receiver calculates the interval between current timer counter and timer counter that stored at packet pair 1 type packet arrival. The acknowledgement of packet with packet pair 2 type option is transmitted immediately. Figure 3 shows example of packet pair 2 format.

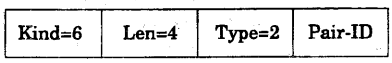

Fig. 3 PacketPair 2 format

(3) Packet Pair reply
The packet pair reply type is to be used by data receiver. The acknowledgement of packet with packet pair 2 type option is transmitted with packet pair reply type option. The packet pair reply type option consists of pair ID and timer value. Pair ID indicate packet pair identificaton. Timer value indicate interval of packet pair. Timer value is difference between timer counter value for packet pair 1 type packet arrival and timer counter value for packet pair 2 type packet arrival. Figure 4 shows example of packet pair reply format.

Kind=6	Len=5	Type=3	Pair-ID	Timer Value

Fig. 4 PacketPair reply format

4.2 Modification of TCP congestion control algorithm

To combine our basic design with TCP, we modified TCP congestion control algorithm.

4.2.1 Slow-Start algorithm

In slow-start phase, window size is increased by 1 segment for each ACK received. In addition, for each new ACK received, opened window size is increased by amount of acknowledged data. Therefore there are always chance of two packets transmission at a time in slow-start phase. The only exception is beginning of slow-start because of window size value 1. Packet pair option is used in this two packets transmission. Figure5 shows example of modified slow-start algorithm.

This figure shows behavior of slow-start algorithm in the networks which has 100msec propagation delay. The bandwidth of bottleneck link is 100Kbps and max segment size is 512byte. In this model, packet pair option is used 0.300msec, 0.540msec, 0.780msec and two consecutive packets transmission can be seen at this time (circled region). The first packet of packet pair is transmitted with packet pair 1 option. The second packet of packet pair is transmitted with packet pair 2 option. The acknowledgement for packet pair is transmitted with packet pair reply option.

The interval of two packets which is expanded through bottleneck link is 40msec. Thus, the timer value of packet pair reply option is 4. When data sender receives packet pair reply option, data sender records timer values in the option. The recorded timer value is used by rate-based transmission control scheme. The packet transmissions at 0.580msec, 0.820msec, 0.860msec, 0.900msec are transmitted by leaky bucket scheme (squared region).

Data receiver uses delayed ack algorithm. The ack transmission at 0.200msec and 0.800msec are transmitted by delayed ack algorithm.

This model shows sender side only probing. This model can estimate only bandwidth of bottleneck link in forward path. Figure 6 shows the model which can estimate bandwidth of bottleneck link in both forward and reverse path.

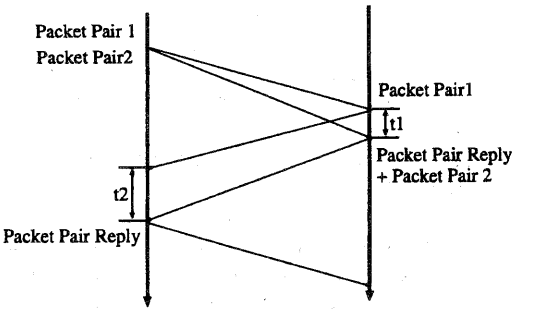

Fig. 6 both sides probing model

As seen in this figure, the acknowledgement for data packet with packet pair 1 option is transmitted immediately with packet pair 1 option. To transmit acknowledge with packet pair option is used for estimation of reverse path's bandwidth. The acknowledgement for data packet with packet pair 2 option is transmitted with packet pair reply option for forward path and packet pair 2 option for reverse path.

If the interval of receiver's packet pair: t2, is almost equals to the interval of sender's packet pair: t1, TCP assumes that this link is symmetric or 'almost symmetric'. 'Almost symmetric link' means the link which bandwidth of forward path is almost same as bandwidth of reverse path. If the interval of receiver's packet pair: t2, is larger than the interval of sender's packet pair: t1, TCP assumes that this link is asymmetric. Usually, delayed ack algorithm transmit acknowledgement every two packet arrival. When new TCP finds an asymmetric link, acknowledgement packet is transmitted every three or four packet arrival in proportion to ratio of t1 to t2. This aggregated acknowledgement causes bursty data transmission in normal TCP. However, the new TCP has traffic shap-

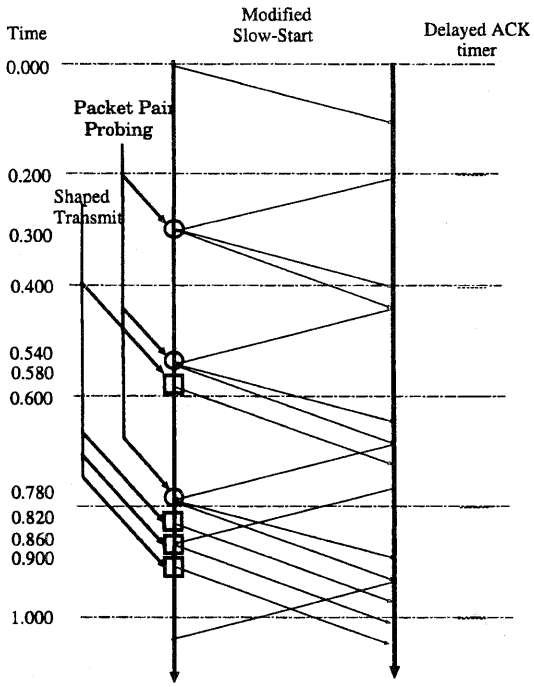

Fig. 5 example of modified slow-start

ing function. Thus the new TCP is not affected by aggregated acknowledgement.

4.2.2 Recovery from packet loss

During the recovery of packet loss, a large ACK may trigger burst transmission[8]. To solve this problem, J.Hoe suggested to limit the number of segments which TCP can inject into the network successively in[8].

The new TCP uses traffic shaping function to solve the problem of burst transmission. After retransmission of lost packets, TCP receives new acknowledgement packets. When new acknowledgement packet is received, The new TCP enters in 'safe-transmit' mode. In safe-transmit mode, TCP transmit packets by using traffic shaping function. The transmission rate in safe-transmit mode is timer values which is acquired by packet pair scheme in slow-start phase.

4.2.3 Congestion avoidance algorithm

When TCP is in congestion avoidance, there is not always chance of two packets transmission at a time. Furthermore, in congestion avoidance phase, self-clocking status may be established.

The goal of new TCP is to establish 'self-clocking' smoothly, not to add any changes to essence of TCP: self-clocking. When TCP is in slow-start phase or recovering from losses, TCP connection does not establish self-clocking status yet. Therefore, to control interval of data transfer in slow-start phase does not destroy essence of TCP. However, if TCP is in self-clocking status, to control interval of data transfer may destroy the status. Therefore, we added no modification to congestion avoidance algorithm.

5. Simulation result

To analyze effect of new TCP scheme, we implemented this algorithm on ns simulator[9] developed by LBL. Our implementation is based on TCP reno[2] module in ns.

5.1 Simulation in symmetric link

Figure 7 shows the network configuration of the analysis of transmission performance in symmetric link. In figure 7, node 1 and 4 are connected by 10Mbps link. The propagation delay of 10Mbps link is set to 5msec. The link between router 2 and 3 is a bottleneck link. This bottleneck link has 500Kbps bandwidth and 100msec propagation delay. The queue size of router is set to 5. In this configuration, node 1 sends data to node 4 over TCP. Data packets and ack packets are sent through same path. The new TCP uses sender side only probing in

this case.

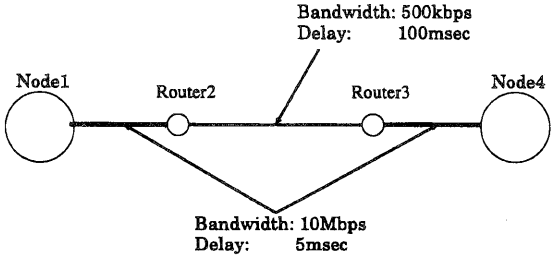

Fig. 7 symmetric network configuration

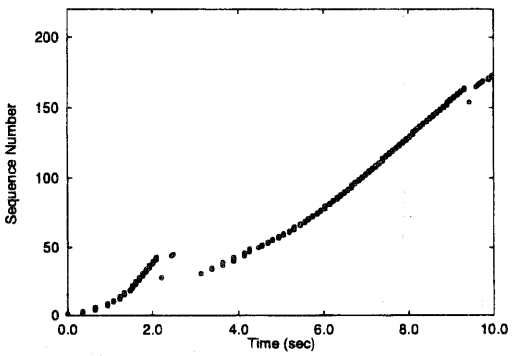

Fig. 8 sequence number variation of normal TCP

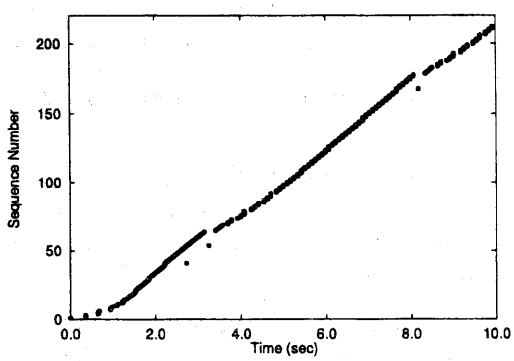

Fig. 9 sequence number variation of new TCP

Figure 8 shows the variation of sequence number by normal TCP algorithm. In this simulation, normal TCP transmits 173 segments in 10 seconds. Figure 9 shows the variation of sequence number by new TCP algorithm. In this simulation, new TCP transmits 212 segments in 10 seconds. Using new TCP algorithm, sequence number is increased slowly in slow-start phase rather than normal TCP algorithm. This is the effect of traffic shaping function. This effects leads smooth transition from slow-start and congestion avoidance.

Compared with two algorithm, 22% improvement of transmission performance is maintained.

5.2 Simulation in asymmetric link

Figure 10 shows the network configuration of the analysis of transmission performance in asymmetric link.

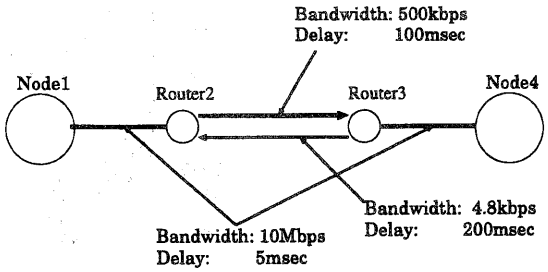

Fig. 10 symmetric network configuration

In figure 10, node 1 and 4 are connected by 10Mbps link. The propagation delay of 10Mbps link is set to 5msec. The link from router 2 to router 3 has 500Kbps bandwidth and 100msec propagation delay. The queue size of router is set to 5. The link from router 3 to router 2 has 4.8Kbps bandwidth and 200msec propagation delay. The queue size of router is set to 5. In this configuration, node 1 sends data to node 4 over TCP. Data packets are sent through 500Kbps bottleneck link. Ack packets are sent through 4.8Kbps bottleneck link.

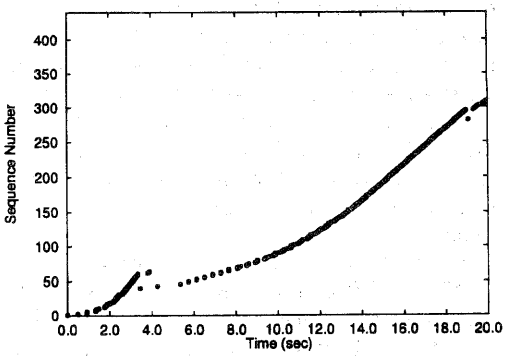

Fig. 11 sequence number variation of normal TCP

Figure 11 shows the variation of sequence number by normal TCP algorithm. In this simulation, normal TCP transmits 308 segments in

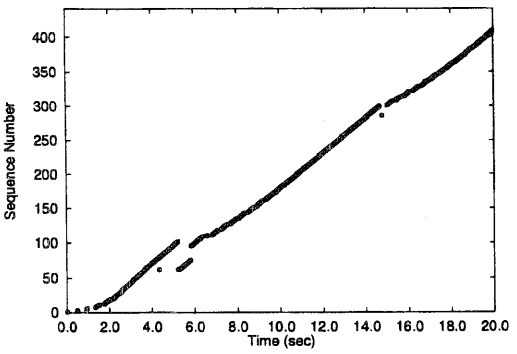

Fig. 12 sequence number variation of new TCP

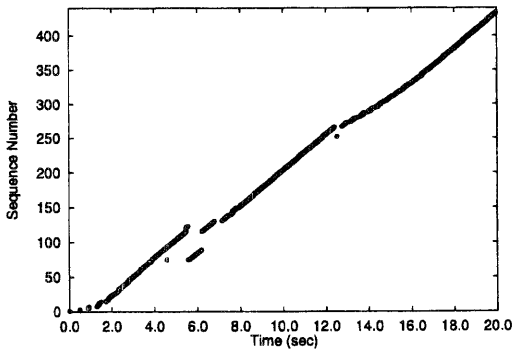

Fig. 13 sequence number variation of new TCP

20 seconds. Figure 12 shows the variation of sequence number by new TCP algorithm with sender side only probing. In this simulation, new TCP transmits 407 segments in 20 seconds. Compared to normal TCP, 32% improvement of transmission performance is maintained.

Figure 13 shows the variation of sequence number by new TCP algorithm with both side probing. In this simulation, new TCP transmits 433 segments in 20 seconds. Compared with normal TCP, 40% improvement of transmission performance is maintained.

6. Conclusion

In this paper, we introduced several techniques for improving TCP performance. We developed new TCP option: packet pair option to estimate bottleneck link's bandwidth. We also developed traffic shaping function to reduce bursty transmission in TCP connection. To combine these techniques with TCP, we modified TCP's congestion control algorithms in some points.

Tested in simulations, these techniques enables TCP to transit from slow-start phase to congestion avoidance phase smoothly. Compared to normal TCP, new TCP including these techniques attain over 20% improvement of transmission performance in symmetric links. In asymmetric links, over 30% performance improvement is maintained by using sender side only probing technique and over 40% improvement is maintained by both side probing technique.

Design of these techniques are simple enough to implement this algorithm on existing operating systems. More analysis of algorithm and extensive experiment over simulation and real internet are being conducted.

Acknowledgments

The author thanks Hiroyuki Kusumoto and Kazunori Sugiura for their detailed and helpful comments and suggestions on this paper. The author also thanks members of WIDE project for their helpful comments on early concept of this paper.

References

1) Van Jacobson.: Congestion Avoidance and Control *Proceedings of ACM SIGCOMM '88*, August 1988.

2) Van Jacobson. Berkeley TCP evolution from 4.3-tahoe to 4.3-reno *Proceedings of 18th IETF*, page 365 1990.

3) Vern Paxson. "End-to-End Internet Packet Dynamics" Proceedings of ACM SIGCOMM '97 June 1997.

4) W. Richard Stevens. : TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms *RFC2001*, January 1997.

5) S. Keshav.: A Control-Theoretic Approach to Flow Control, *Proceedings of SIGCOMM '91*, pp. 3-15, September 1991.

6) J-C. Bolot. : End-to-End Packet Delay and Loss Behavior in the Internet, *Proceedings of SIGCOMM '93*, pp. 289-298, September 1993.

7) R. Carter and M. Crovella,: Measuring Bottleneck Link Speed in Packet-Switched Networks, *Technical Report BU-CS-96-006, Computer Science Department, Boston University*, March 1996.

8) Janey C. Hoe.: Improving the Start-up Behavior of a Congestion Control Scheme for TCP *Proceedings of ACM SIGCOMM '96*, August 1996.

9) Sally Floyd. Simulator Test. *Technical report*, July 1995.