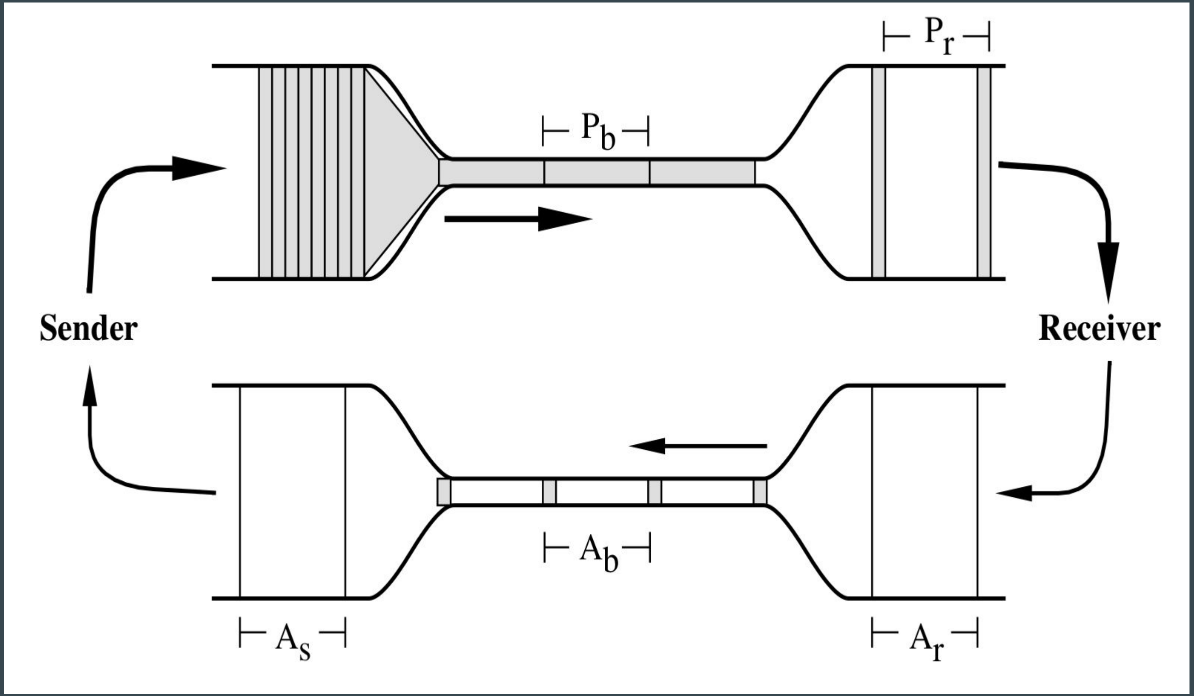
Congestion Avoidance and Control

Rohan Tabish and Zane Ma

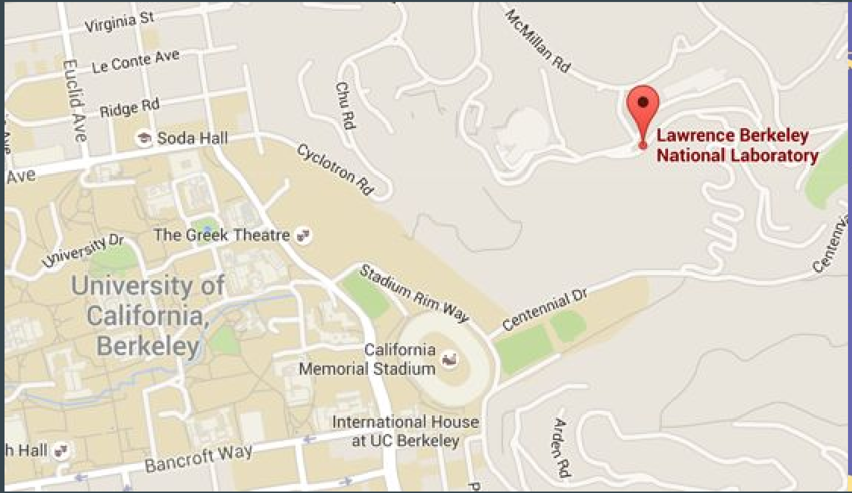
TCP is self-clocking

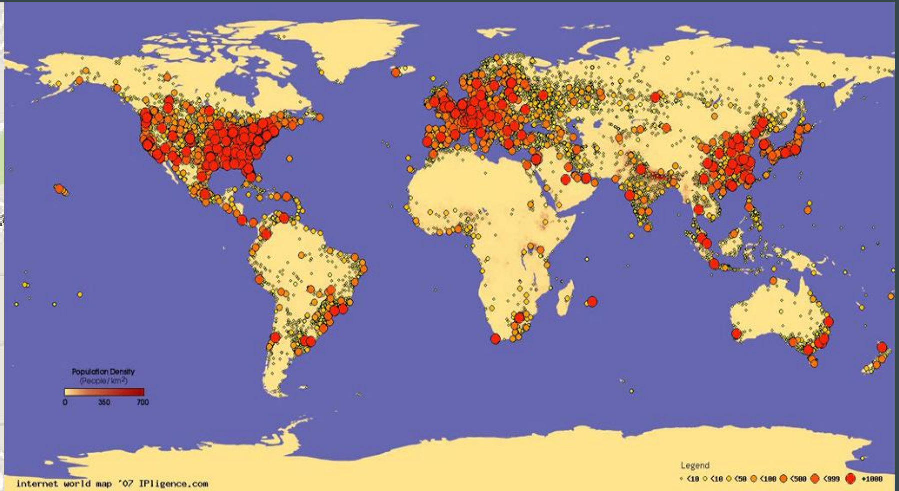
- Self-clocking systems should be robust



Congestion collapse

- Internet had first of what became a series of congestion collapses in **October 1986**.
  - Data throughput from Lawrence Berkeley Lab (LBL) to UC Berkeley dropped **1000x** from 32kbps to 40bps.

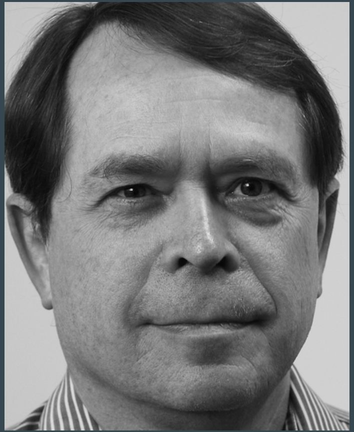




Why does congestion exist?

- Limited **bandwidth**, **buffers** and **queues**.
- Network layer **gives up** in the face of adversity
- End hosts re-try or new hosts join and further overwhelm

Jacobson to the Rescue



- Growth - getting to equilibrium

- Maintenance - **packet conservation**

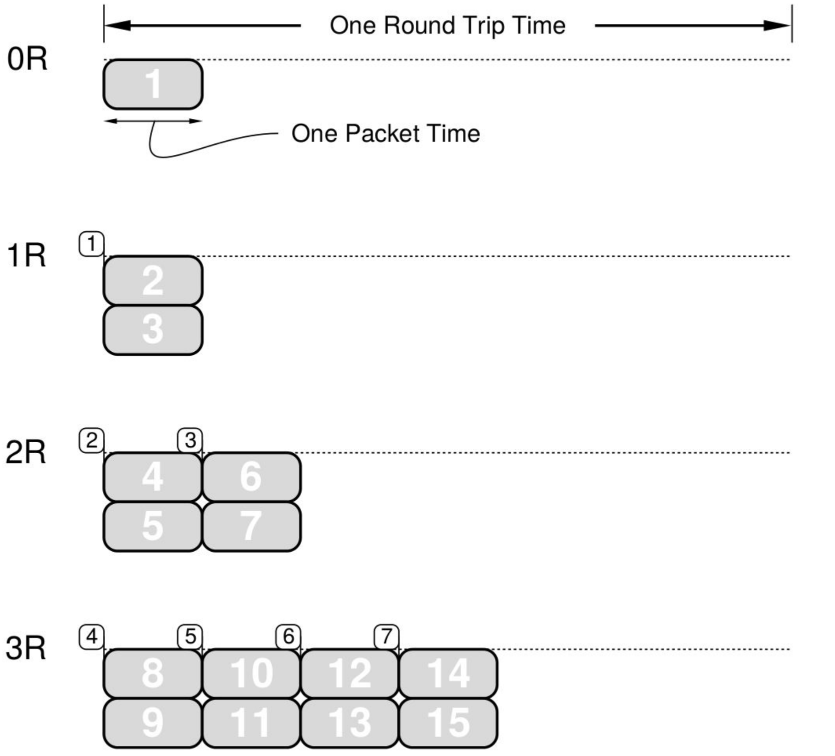
  “A new packet isn’t put into the network until an old packet leaves”

- Retreat - leaving the network

Not-so-slow start

- Relatively slow, but still exponentially fast!
- cwnd (congestion window) that increases by 1 for every ACK
- continue until packet loss, full receiver window, or slow start threshold reached

Figure 2: The Chronology of a Slow-start



Getting to Equilibrium Using Slow-start Algorithm

Figure 3: Startup behavior of TCP without Slow-start

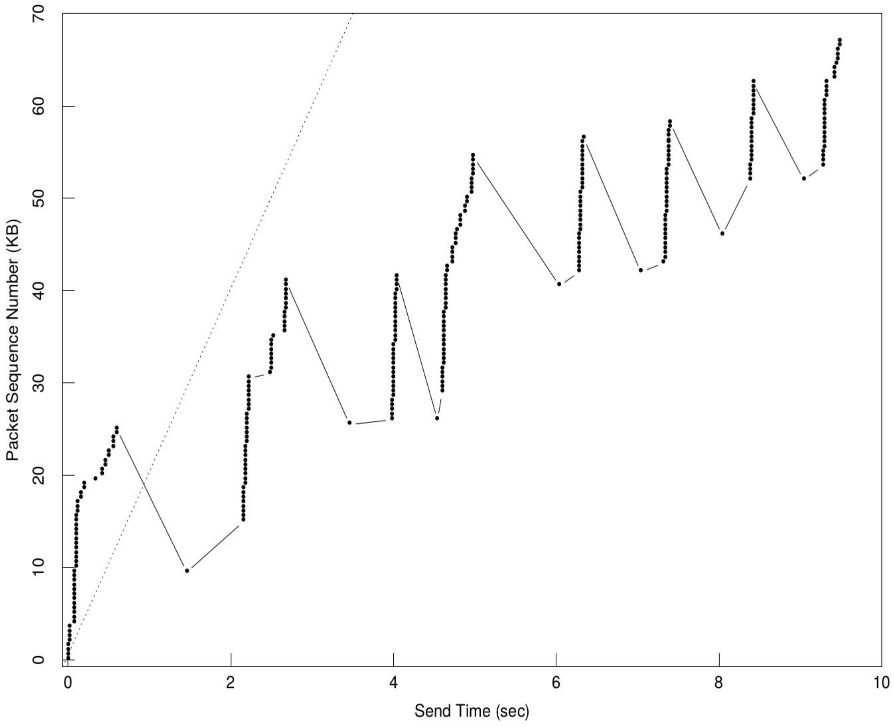
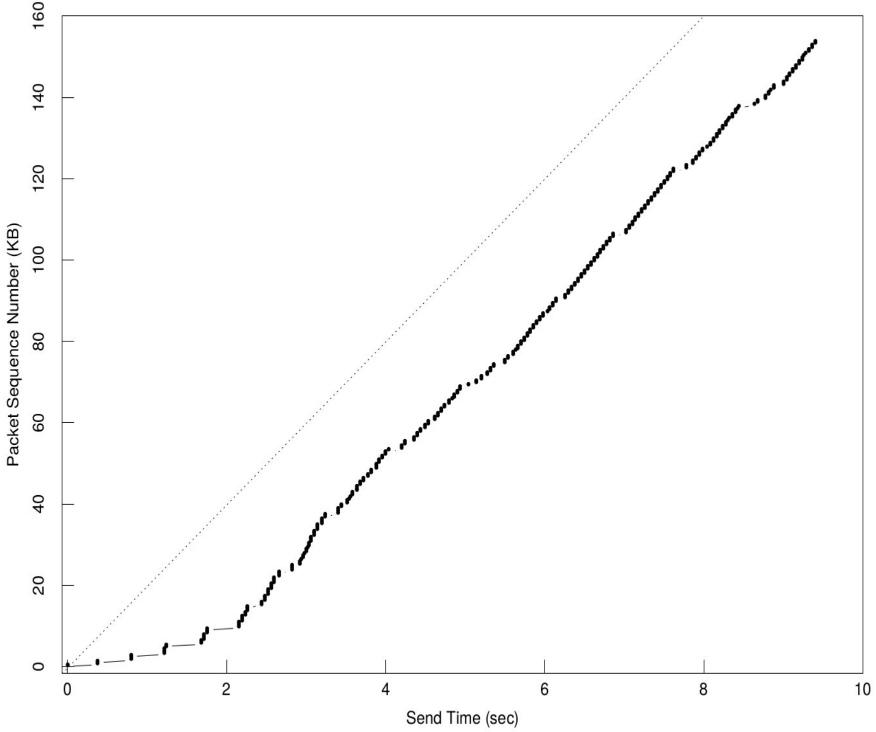


Figure 4: Startup behavior of TCP with Slow-start



Equilibrium Maintenance

- Challenge: adapt quickly to equilibrium changes
- Respond quickly to packet loss due to congestion - retransmit timer
- Maximize the amount of bandwidth used without overwhelming network

Conservation at equilibrium: round-trip timing

- The TCP protocol specification suggests estimating mean round trip time via the low-pass filter is given by

$$R \leftarrow \alpha R + (1 - \alpha)M$$

where,

R = average.

RTT = estimate.

M = RTT measurement recently acked data packet.

α = filter gain constant suggested value of 0.9

- Once the R estimate is updated, the retransmit timeout interval, rto, for the next packet sent is set to βR.
- The parameter β accounts for RTT **variance** - should be dynamic!

Conservation at equilibrium: round-trip timing

Figure 5: Performance of an RFC793 retransmit timer

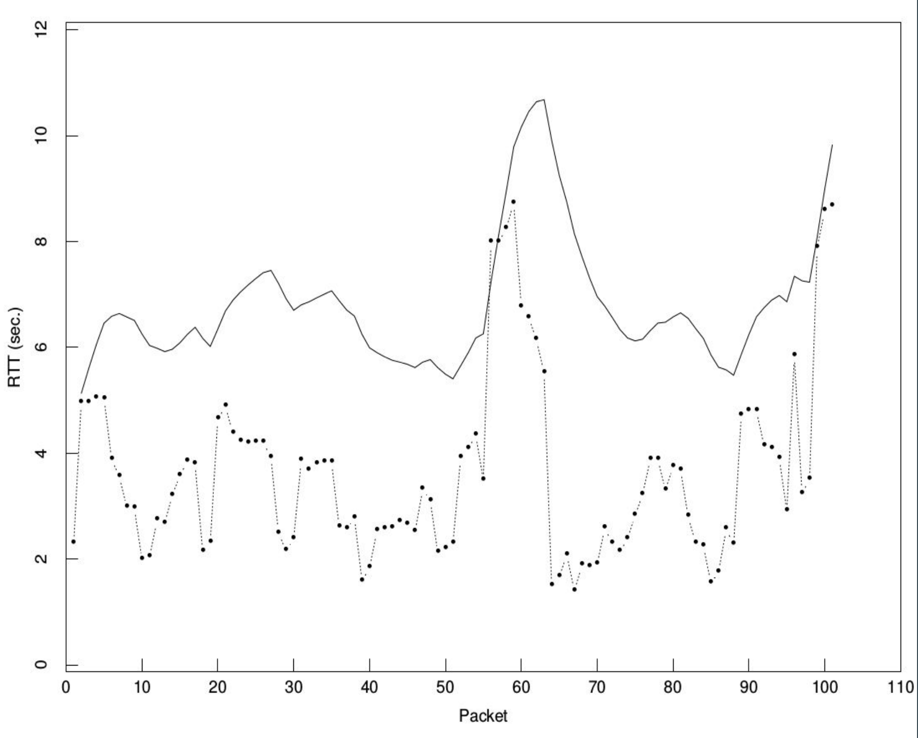
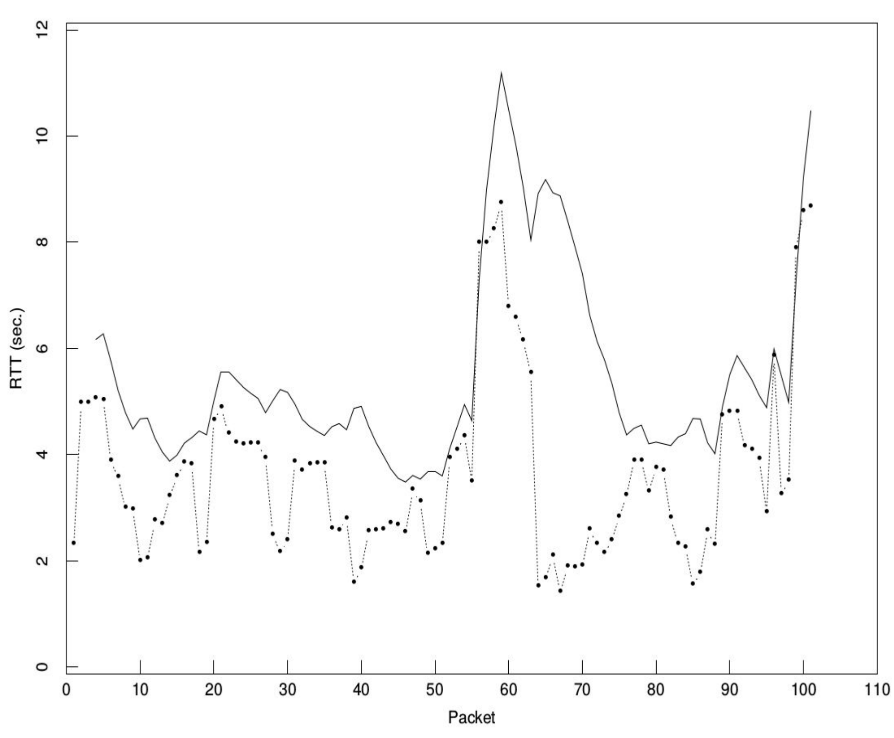


Figure 6: Performance of a Mean+Variance retransmit timer



Adapting to the path: congestion avoidance

- If timers are in good shape, then timeout happens because of packet loss

- Packet loss happens because of two reasons:
  - They are damaged in transit.
  - The network is congested and there is insufficient buffer capacity.

- Loss due to damage less than **one percent**. Most loss is due to **congestion**.

Adapting to the path: congestion avoidance

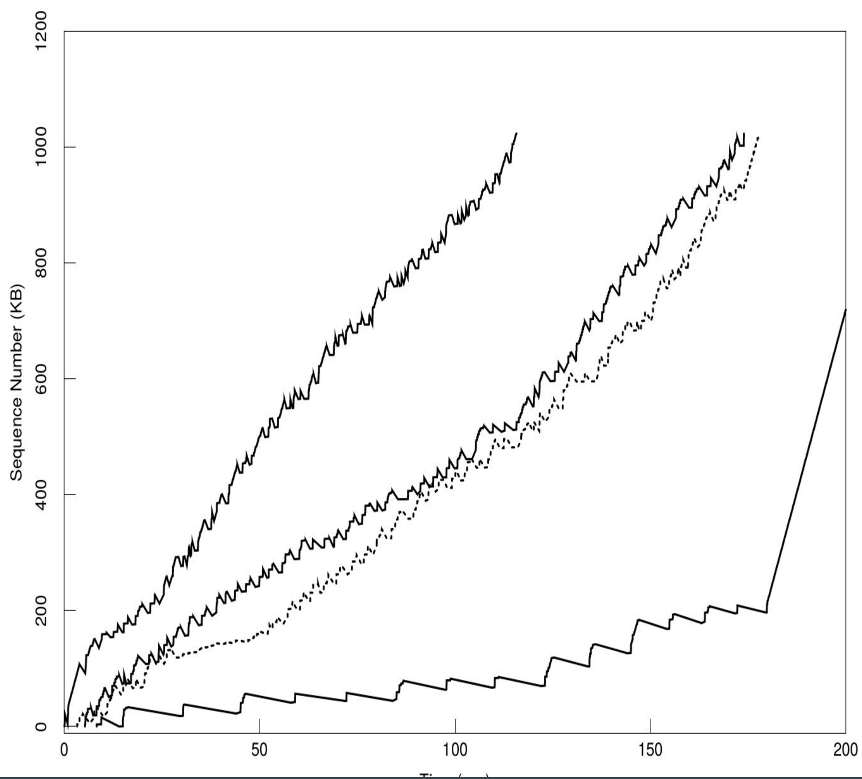
- The congestion avoidance strategy has two components:
  - Detection of congestions - How?
  - Reduce network usage

- The congestion avoidance algorithm works as follows:
  - On any timeout, set cwnd to half the current window size (multiplicative decrease)
  - On each ack for new data, increase cwnd by 1/cwnd (additive increase).
  - When sending, send the minimum of the receiver's advertised window and cwnd.

No Congestion Avoidance

- 4,000 of 11,000 packets sent were retransmissions (i.e., half the data packets were retransmitted)
- Since the link data bandwidth is 25 KBps, each of the four conversations should have received 6 KBps. Instead, one conservation got 8 KBps, two got 5 KBps, one got 0.5 KBps and 6 KBps has vanished

Figure 8: Multiple, simultaneous TCPs with no congestion avoidance

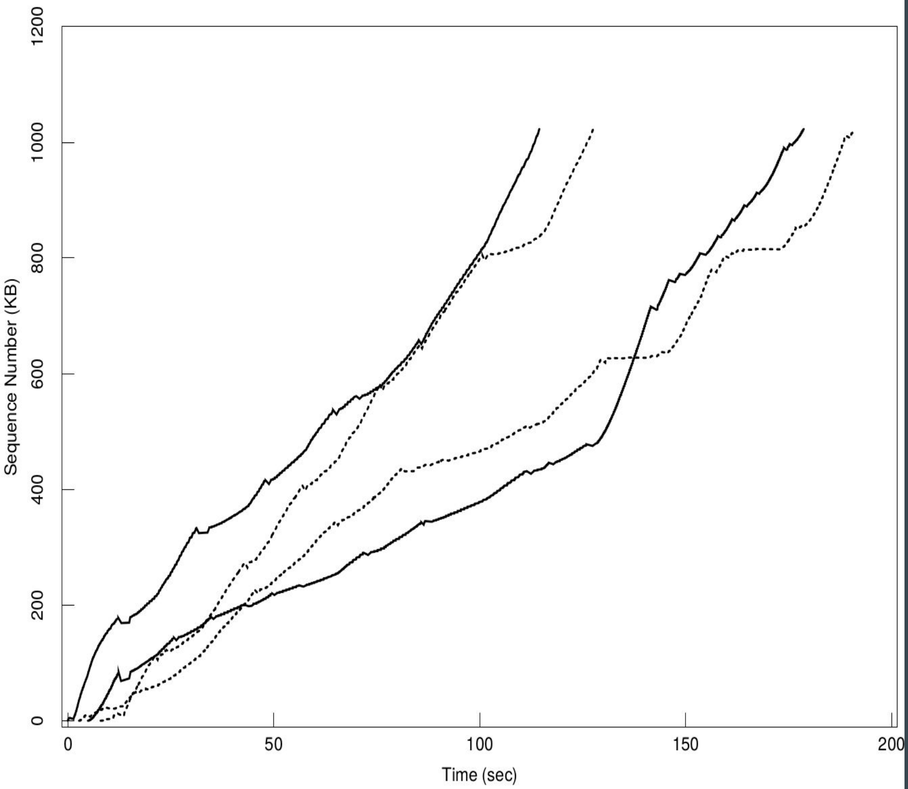


Congestion Avoidance

- 89 of 8281 packets sent were retransmissions.

- Two of the conversations got 8 KBps and two got 4.5 KBps.

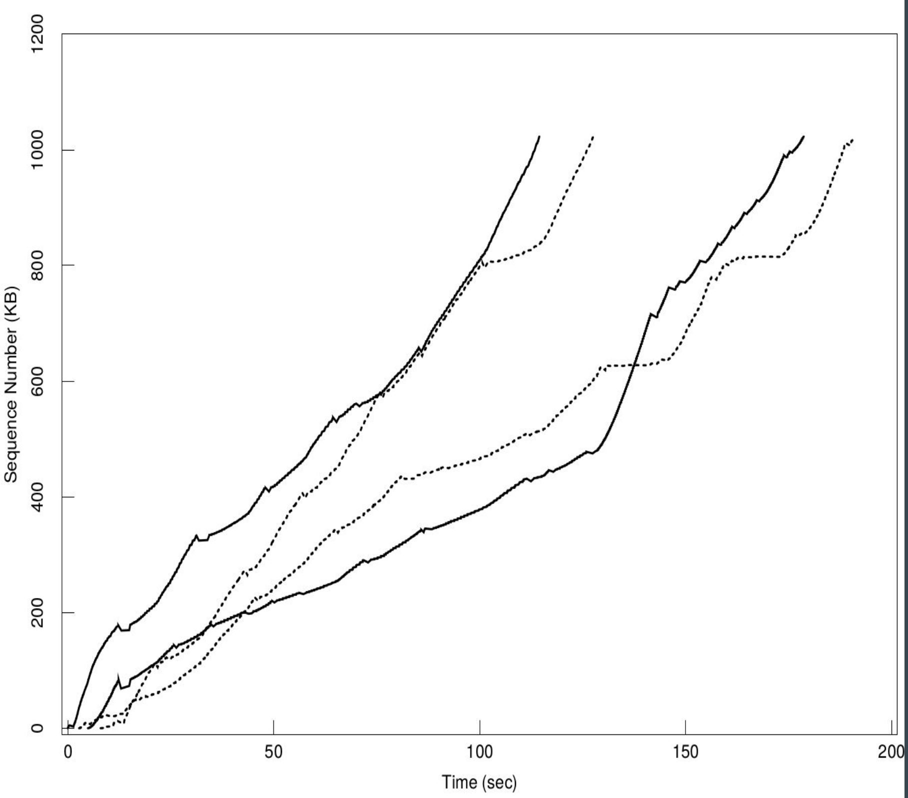
Figure 9: Multiple, simultaneous TCPs with congestion avoidance



Congestion Avoidance

❖ The difference between the high and low bandwidth senders was due to the receivers. The 4.5 KBps senders were talking to 4.3 BSD receivers which would delay an ack until 35% of the window was filled or 200 ms had passed. This meant the sender would deliver bursts of 5–7 packets on each ack.

Figure 9: Multiple, simultaneous TCPs with congestion avoidance

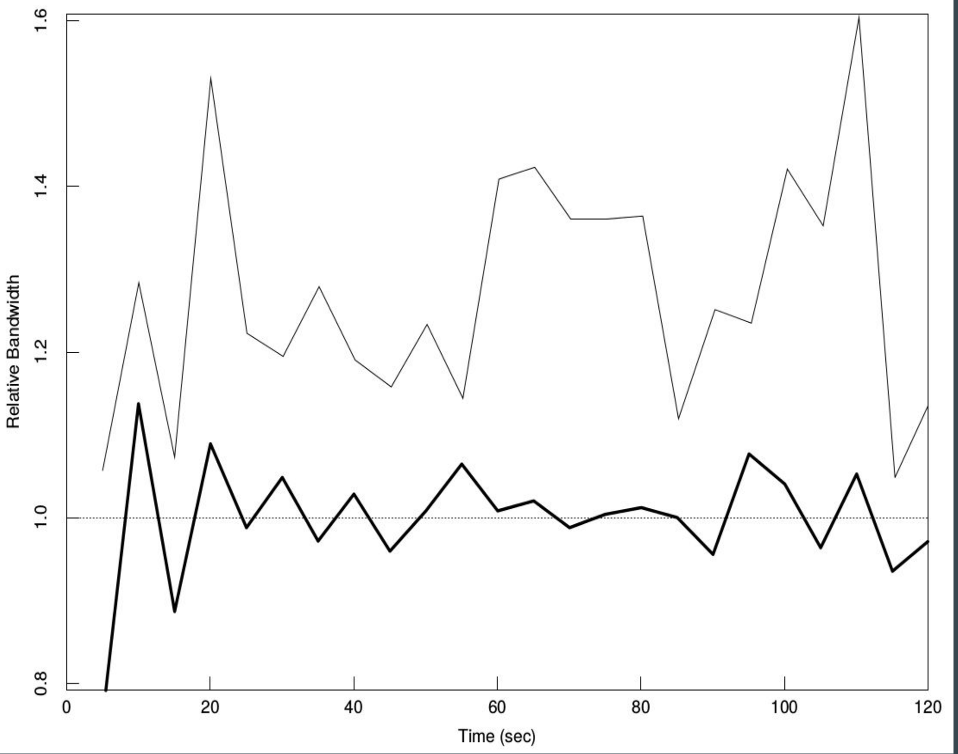


Congestion Avoidance Bandwidth

❖ Thin Line is old TCP with congestion control which is using more BW and it's not in equilibrium.

❖ Thick line is new TCP with congestion control which is not exceeding available BW.

Figure 10: Total bandwidth used by old and new TCPs

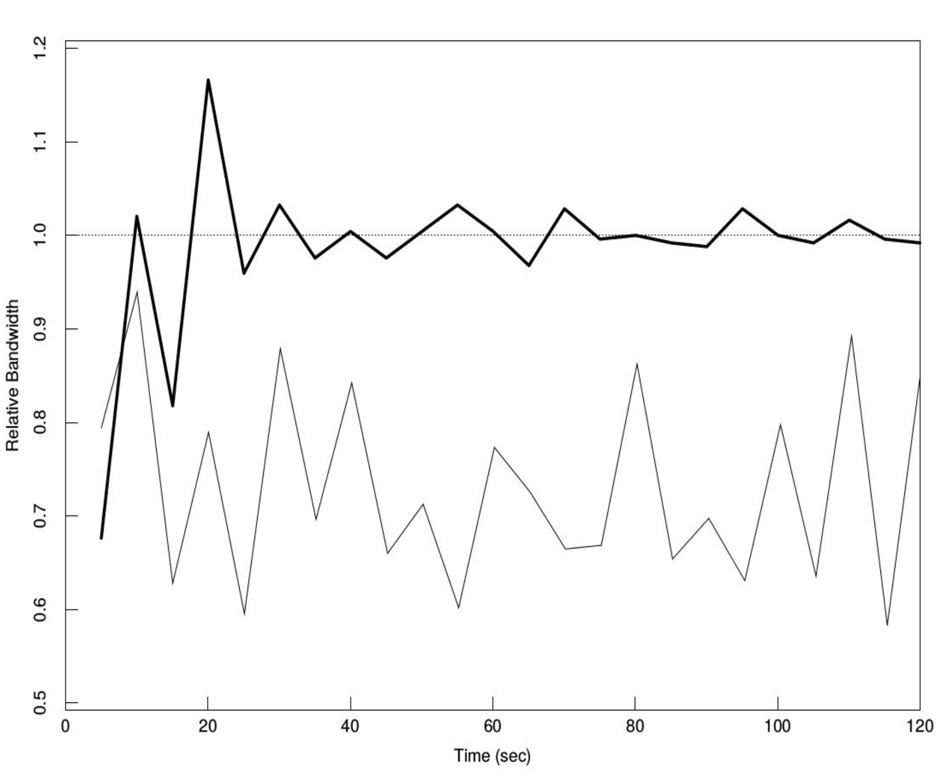


Congestion Avoidance Effective BW

- ❖ Thin Line is old TCP which is using 75% of the BW.

- ❖ Thick line is new TCP which is using almost 100% of the BW.

Figure 11: Effective bandwidth of old and new TCPs



Summary

- There are three failures of packet conservation principle that can be avoided using Slow-start, Round-Trip Timing and Congestion Control algorithm.

- Be conservative around timers and failure

Discussion

- What are some alternatives to using RTT-based retransmission timers as a way to detect packet loss? What are some ways to distinguish between different causes of packet loss?
- How does QoS affect congestion avoidance and control? Does congestion avoidance/control become more or less relevant in light of QoS?