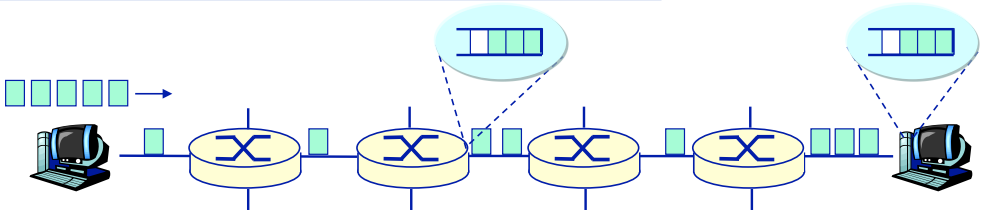
CS 455/555
Intro to Networks and Communications

The Transport Layer
Congestion control in TCP

Dr. Michele Weigle
Department of Computer Science
Old Dominion University
mweigle@cs.odu.edu

http://www.cs.odu.edu/~mweigle/CS455-S13/

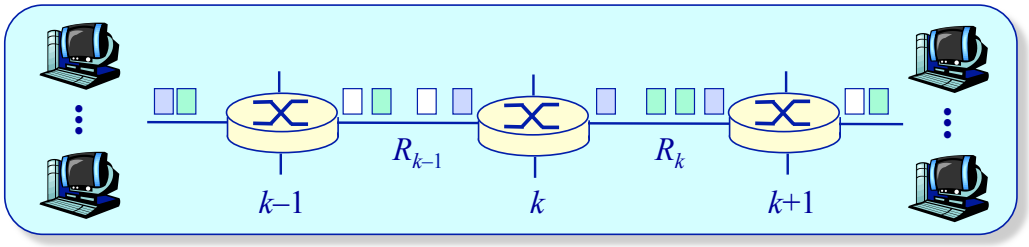
Congestion Control
Congestion control *v.* Flow control



- In *flow control* the sender adjusts its transmission rate so as not to overwhelm the receiver
  - » One source is sending data too fast for a receiver to handle
- In *congestion control* the sender(s) adjust their transmission rate so as not to overwhelm routers in the network
  - » Many sources independently work to avoid sending too much data too fast for the network to handle
- Symptoms of congestion:
  - » Lost packets (buffer overflow at routers)
  - » Long delays (queuing in router buffers)

Congestion Control

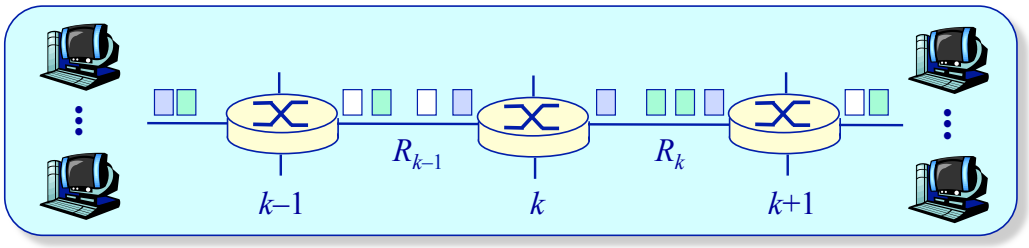
Fairness



- When a connection slows down, by how much should it slow down?
- If n_k connections share a congested link k with capacity R_k, each connection should receive $r = R_k/n_k$ bandwidth
- But what if a connection can't consume R/n bandwidth?

Congestion Control

Fairness



- A connection can't consume more bandwidth on link k than it consumes on any previous link
- If a connection traverses L links then its end-to-end bandwidth is $r \leq \text{MIN}(R_1/n_1, \ldots, R_L/n_L) \leq R_k/n$
- *Fairness* implies that if there exists a connection such that $r \leq R_k/n$, then the connection's unused share of the bandwidth on link k, $R_k/n - r$, is evenly shared with all other connections that are capable of consuming more bandwidth

Congestion Control

MAX-MIN Fairness

- Consider a set of n connections that consume

$$r_1 \leq r_2 \leq \ldots \leq r_n$$

bits per second of bandwidth

- "Fairness" implies that…
  - » No connection receives more bandwidth than it requires
  - » If a connection receives less bandwidth than it requires then it receives the same amount of bandwidth as all other unsatisfied connection

Initially each connection gets R/n of a link's capacity. If $r_1 < R/n$ then the unused $R/n - r_1$ is reallocated such that flows 2 through n receive

$$R/n + \frac{R/n - r_1}{n-1}$$

of the link's capacity.

Congestion Control

MAX-MIN Fairness

- Consider a set of n connections that consume

$$r_1 \leq r_2 \leq \ldots \leq r_n$$

bits per second of bandwidth

- "Fairness" implies that…
  - » No connection receives more bandwidth than it requires
  - » If a connection receives less bandwidth than it requires then it receives the same amount of bandwidth as all other unsatisfied connection

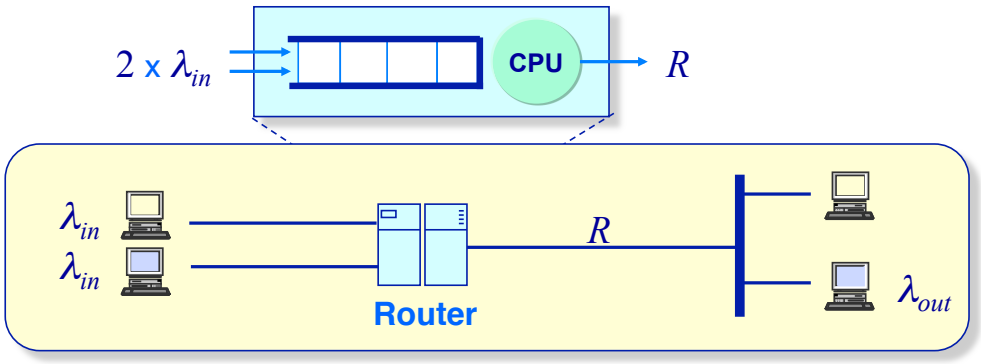
Initially each connection gets R/n of a link's capacity. If $r_1 < R/\mathrm{n}$ and $r_2 < R/n + (R/n - r_1)/(n-1)$ then the unused bandwidth is reallocated such that flows 3 through n receive

$$R/n + \frac{R/n - r_1}{n-1} + \frac{R/n + (R/n - r_1)/(n-1) - r_2}{n-2}$$

of the link's capacity.

The Causes and Effects of Congestion

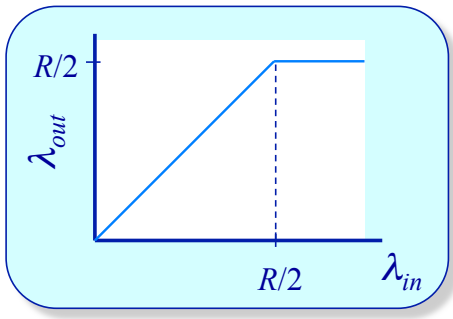
Scenario 1: Two equal-rate senders share a single link



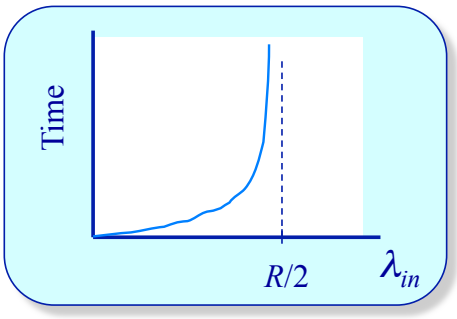
- Two sources send at an average rate of λ_{in} to two receivers across a shared link with capacity R
  - » Data is delivered to the application at the receiver at rate λ_{out}
- Packets queue at the router
  - » Assume the router has infinite storage capacity (Thus no packets are lost and there are no retransmissions)

The Causes and Effects of Congestion

Scenario 1: Two equal-rate senders share a single link



Throughput

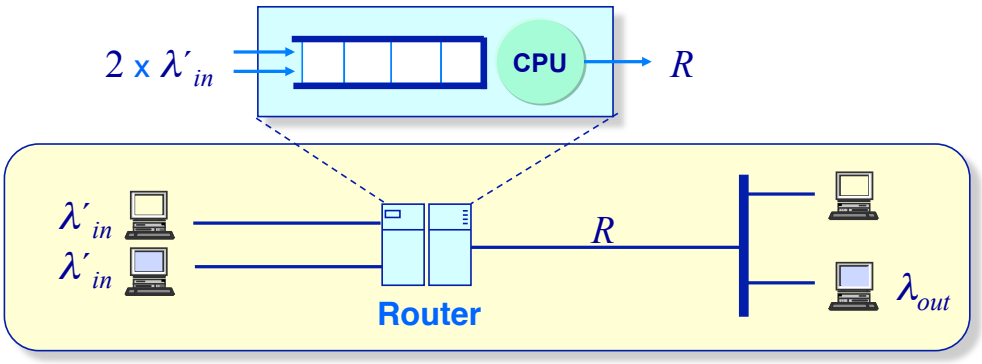


Delay

- The maximum achievable per connection throughput is constrained by $^1/_2$ the capacity of the shared link
- Exponentially large delays are experienced when the router becomes congested
  - » The queue grows without bound

The Causes and Effects of Congestion

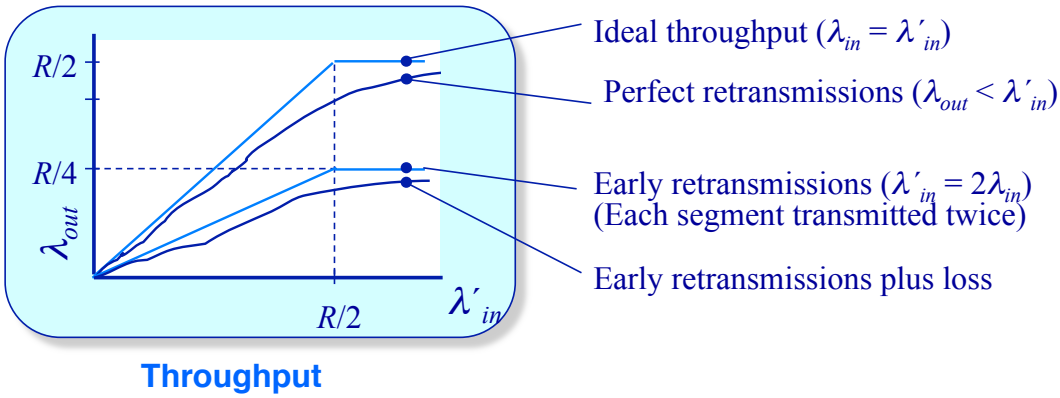
Scenario 2: Finite capacity router queue



- Assume packets can now be lost
  - » Sender retransmits upon detection of loss
- Define *offered load* as the original transmissions plus retransmissions
  - » $\lambda'_{in} = \lambda_{in} + \lambda_{retransmit}$

The Causes and Effects of Congestion

Scenario 2: Throughput analysis

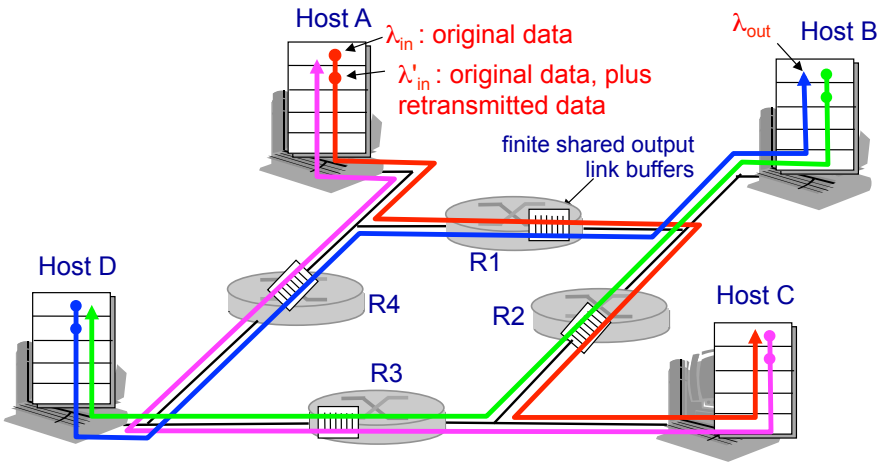


- By definition $\lambda_{out} = \lambda_{in}$
- Retransmission scenarios:
  - » "Perfect" — Retransmissions occur only when there is loss
  - » Early — Delayed packets are retransmitted

The Causes and Effects of Congestion

Scenario 3: Multihop paths

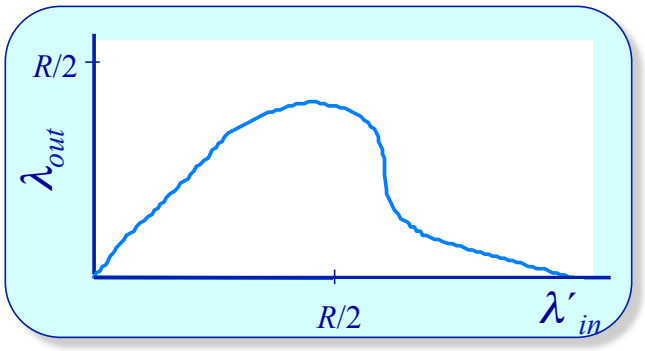
Four senders, four routers, two-hop paths



What happens as λ_{in} and λ'_{out} increase?

The Causes and Effects of Congestion

Scenario 3: Throughput analysis



Throughput

- Congestion collapse
  - » All the links are fully utilized but no data is delivered to applications!

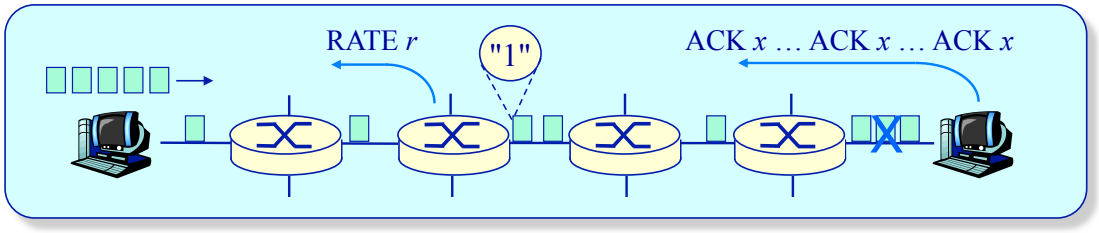
The Causes and Effects of Congestion

Costs of Congestion

- Large queuing delays
- Retransmissions
- Wasted router resources due to forwarding unneeded copies of a packet
- Wasted router resources due to forwarding packets that will be dropped late

Approaches to Congestion Control

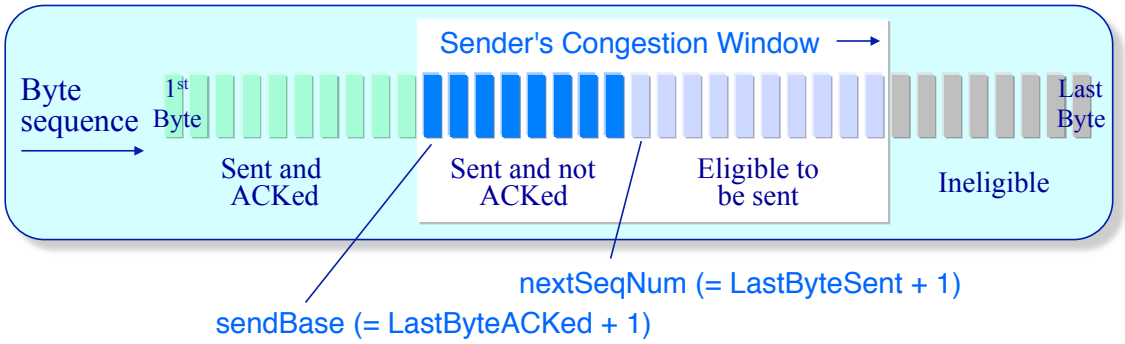
End-to-end *v.* Hop-by-hop



- End-to-end congestion control
  - » End-systems receive no feedback from network
  - » Congestion inferred by observing loss and/or delay
- Hop-by-hop congestion control
  - » Routers provide feedback to end systems
    - ❖ Network determines an explicit rate that a sender should transmit at
    - ❖ Network signals congestion by setting a bit in a packet's header (SNA, DECbit, TCP/IP ECN, ATM)

End-to-End Congestion Control

TCP Congestion Control



- Transmission rate is limited by the *congestion window* size, cwnd

```
LastByteSent - LastByteACKed ≤ MIN(cwnd,RcvWindow)
```

- Maximum rate is *w* MSS byte segments sent every RTT

$$throughput = \frac{w \times MSS}{RTT} \quad bytes/sec$$

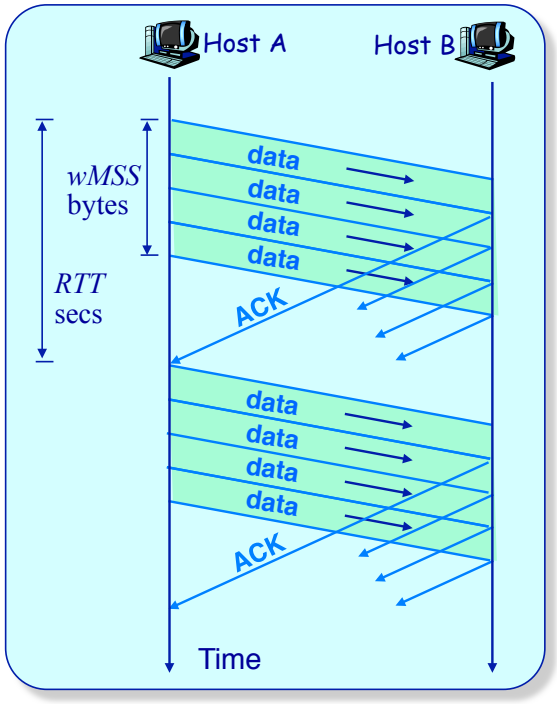
TCP Congestion Control

Congestion window and transmission rate

- If $w \times MSS/R < RTT$, then the maximum rate at which a TCP connection can transmit data is

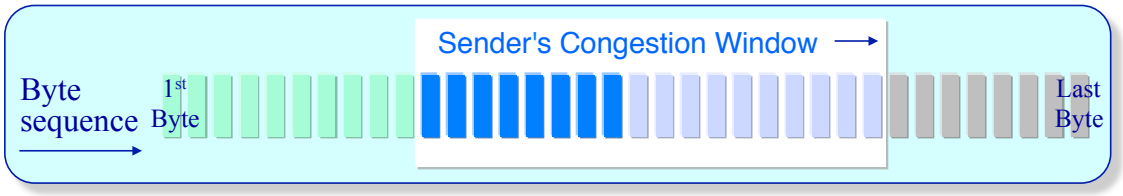
$$\frac{w \times MSS}{RTT} \quad bytes/sec$$

- *w* is the minimum of the number of segments in the receiver's window or the congestion window



TCP Congestion Control

Congestion window control



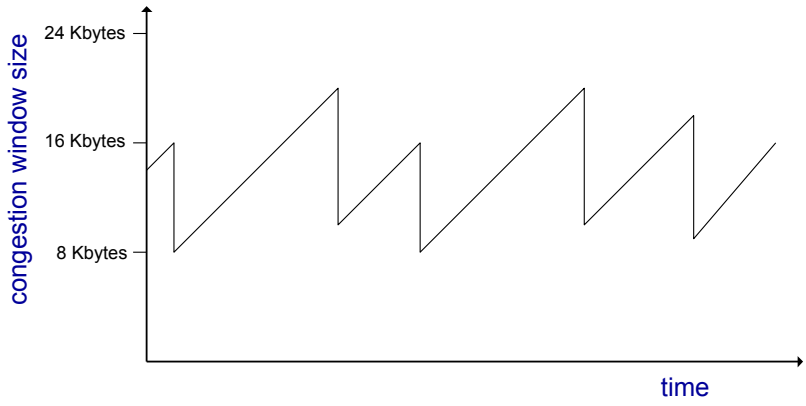
- TCP connections probe for available bandwidth
  - » Increase the congestion window until loss occurs
  - » When loss is detected decrease window, then begin probing (increasing) again
- The congestion window grows in two phases:
  - » *Slow start* — Ramp up transmission rate until loss occurs
  - » *Congestion avoidance* — Keep connection close to sustainable bandwidth
- A window size threshold (bytes transmitted) distinguishes between slow start and congestion avoidance phases

TCP Congestion Control

Additive increase, multiplicative decrease (AIMD)

- *Approach:* increase transmission rate (window size), probing for usable bandwidth, until loss occurs
  - » *additive increase:* increase **cwnd** by 1 MSS every RTT until loss detected
  - » *multiplicative decrease*: cut **cwnd** in half after loss

Saw tooth behavior: probing for bandwidth



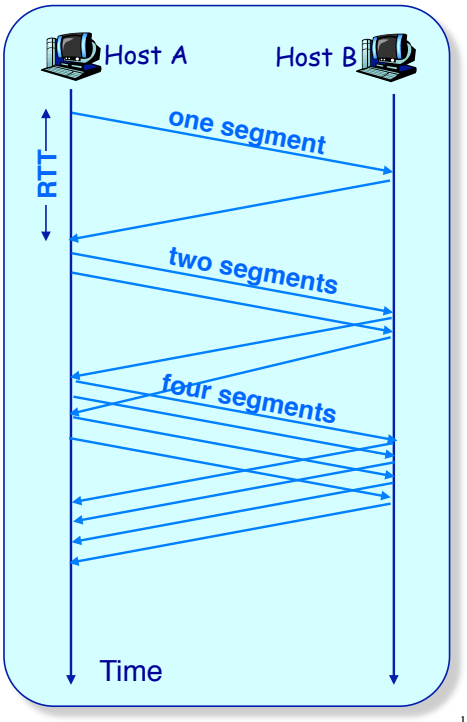
TCP Congestion Control

Slowstart

```
cwnd = 1 MSS
for (each original ACK received)  cwnd++
   until (loss event OR cwnd > threshold)
```

- Exponential increase in window size each RTT until:
  - » Loss occurs
  - » *cwnd = threshold*

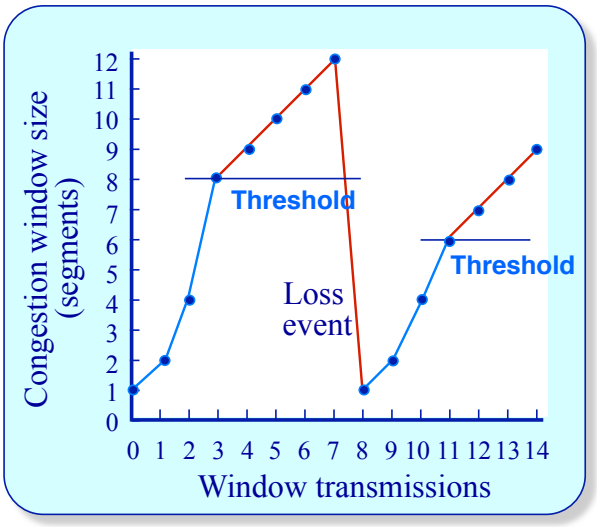
  (Not so slow!)



TCP Congestion Control

Congestion avoidance

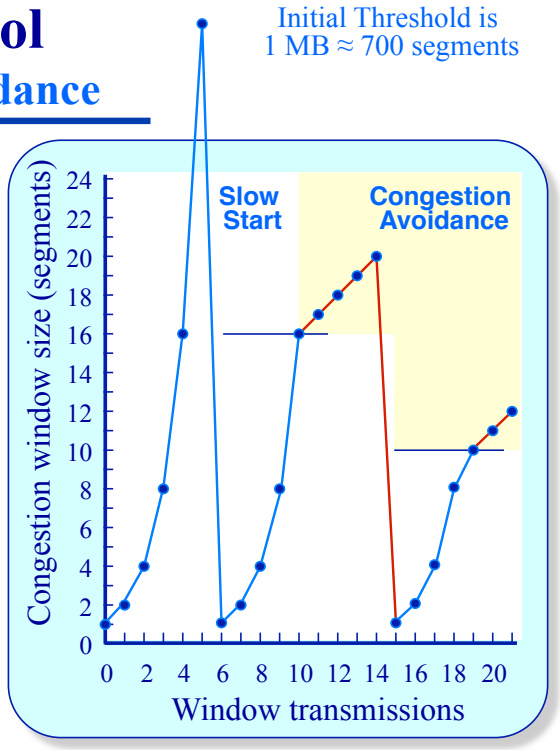
```
/* slowstart is over;
   cwnd > threshold
*/
until (loss event) {
  whenever cwnd segments
     ACKed:
        cwnd++
  }
/* loss event timeout */
threshold = cwnd/2
cwnd  = 1 MSS
perform slowstart
```



TCP Congestion Control

Slow-start *v*. Congestion avoidance

- The threshold is an estimate of a "safe" level of throughput that is sustainable in the network
  - » The threshold specifies a throughput that was sustainable in the recent past
- Slow-start quickly increases throughput to this threshold
- Congestion avoidance slows probes for additional available bandwidth beyond the threshold

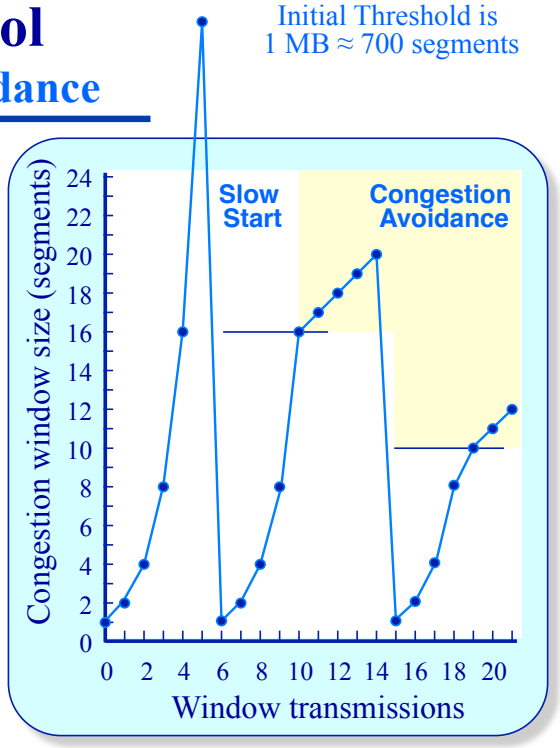


Assume $RTT > \frac{w \times MSS}{R}$

TCP Congestion Control

Slow-start *v*. Congestion avoidance

- Loss (at any time) reduces the "safe" throughput estimate to 1/2 of the current throughput
  - » This is the throughput that resulted in loss
- Slow-start begins anew whenever there is loss
- Throughput at initial threshold = 1 MB/*RTT*
  - » At 1st threshold: 16*MSS*/*RTT*
  - » At 2nd threshold: 10*MSS*/*RTT*

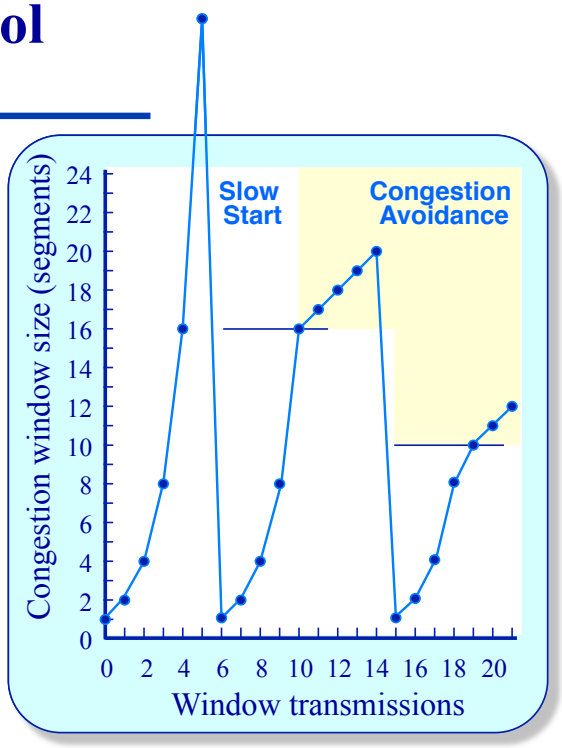


Assume $RTT > \frac{w \times MSS}{R}$

TCP Congestion Control

Major TCP variants

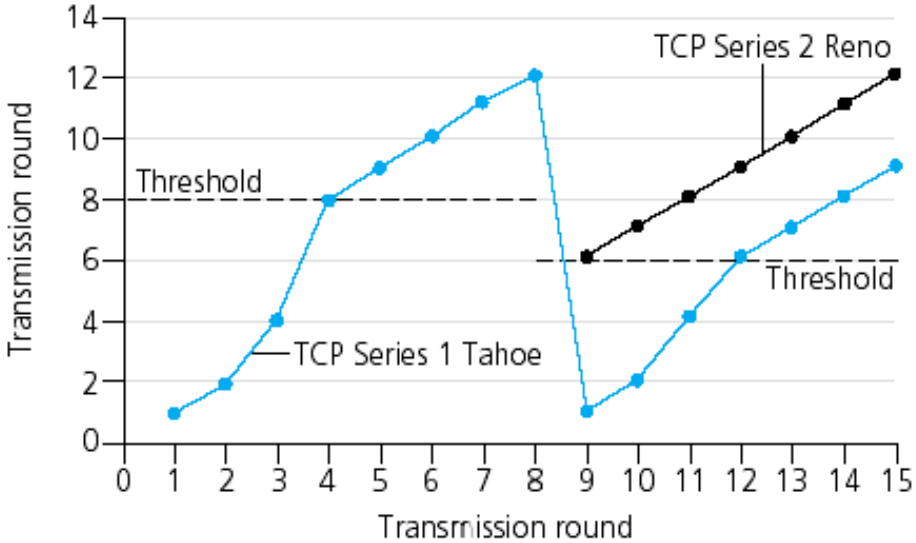
- TCP Tahoe:
  - » Loss signaled by timeout
  - » *threshold* = *cwnd*/2
  - » *cwnd* = 1 MSS
  - » "Fast retransmit"
    - ❖ receipt of 3 duplicate ACKs also signals a packet loss
- TCP Reno:
  - » "Fast recovery"
    - ❖ skips slowstart and continues in congestion avoidance
    - ❖ *cwnd* = *cwnd*/2
    - ❖ additive increase, multiplicative decrease (AIMD)
- Others: TCP NewReno, SACK, …



Assume $RTT > \frac{w \times MSS}{R}$

TCP Congestion Control

Tahoe vs. Reno



TCP Congestion Control
Summary

Goal: Efficient transfer without overwhelming the network

- 2 phases:
  - » *slow-start*
    - each ACK, cwnd++ (each RTT, cwnd doubles)
  - » *congestion-avoidance*
    - each ACK, cwnd += 1/cwnd (each RTT, cwnd++)

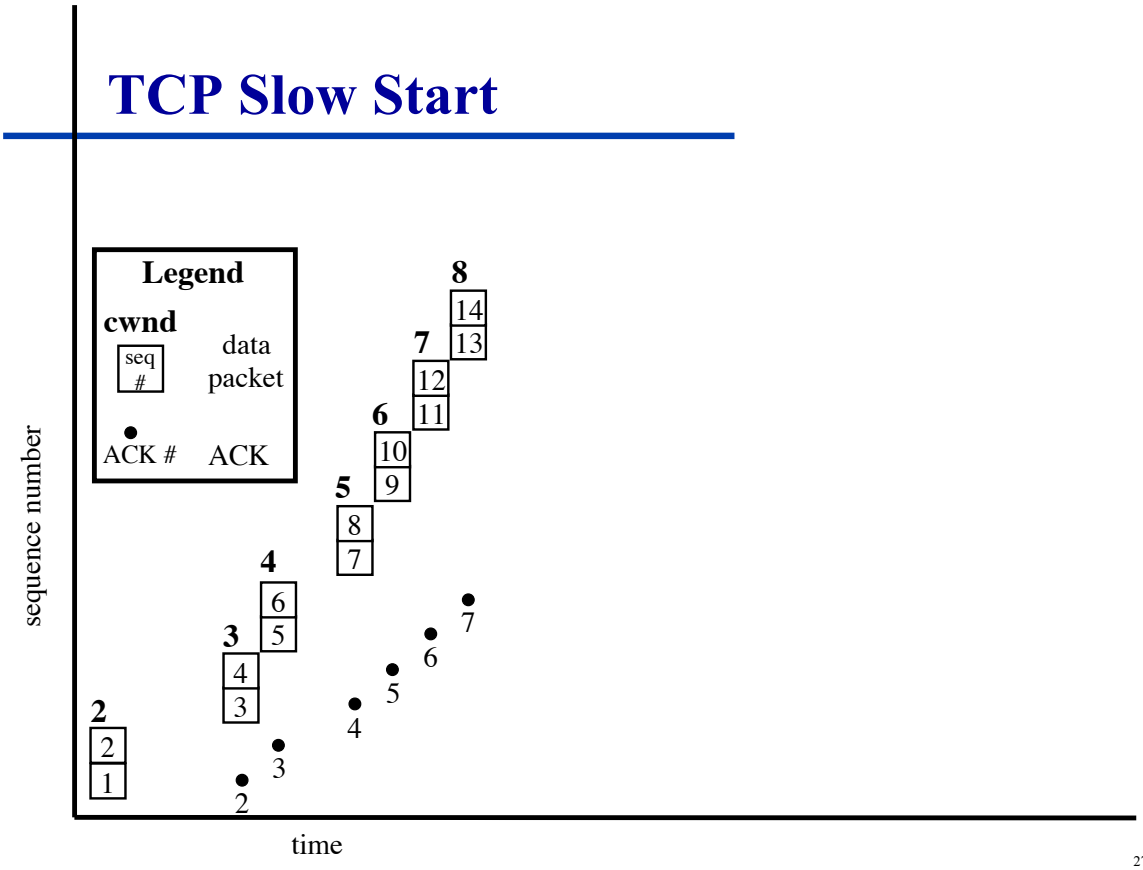
- Control:
  - » if cwnd < ssthresh, *slow-start*
  - » if cwnd >= ssthresh, *congestion-avoidance*

TCP Congestion Control
Summary

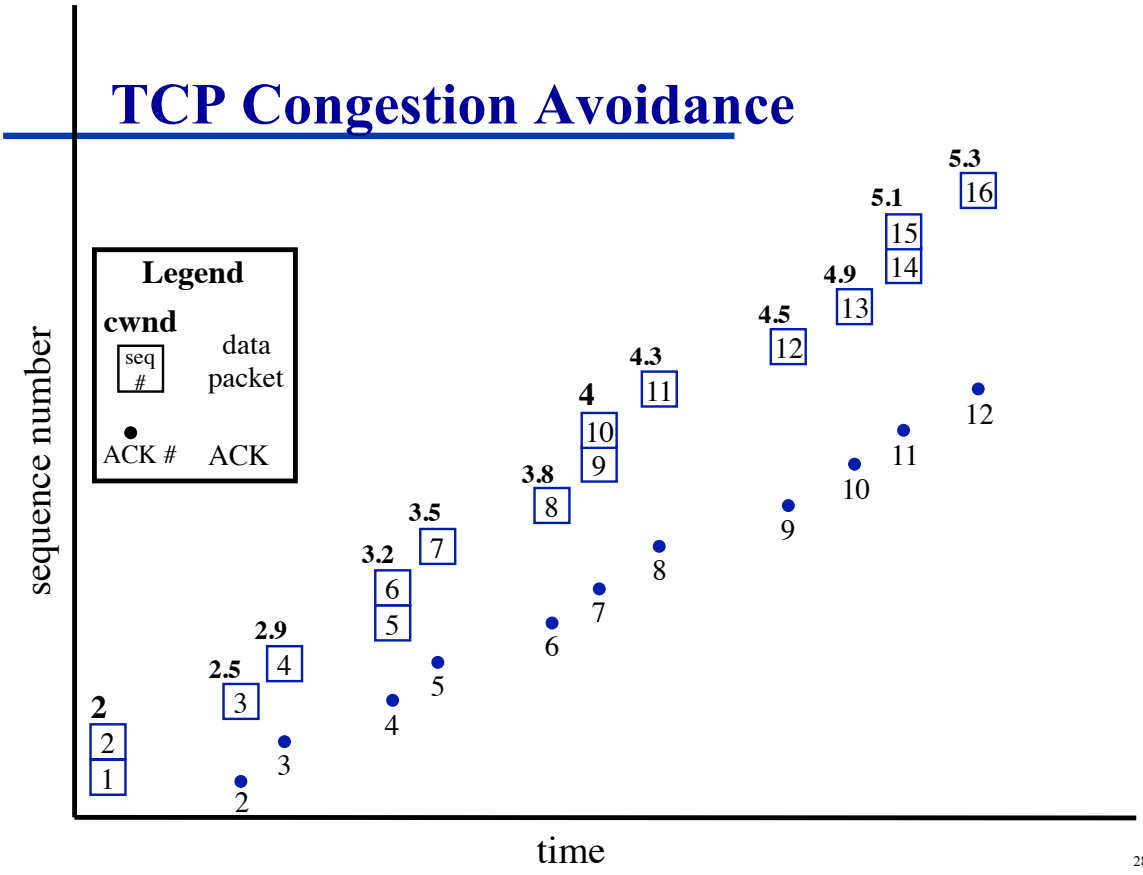
- Loss:
  - » timeout
    - ssthresh = 1/2 cwnd
    - cwnd = 1
  - » 3 duplicate ACKs (*fast retransmit*)
    - ssthresh = 1/2 cwnd
    - *TCP Tahoe*: cwnd = 1
    - *TCP Reno*: cwnd = 1/2 cwnd (*fast recovery)*

- Other Points:
  - » cwnd is only reduced when loss is inferred
  - » a lost packet is retransmitted before cwnd is reduced
  - » if RTT is stable, cwnd controls the sending rate

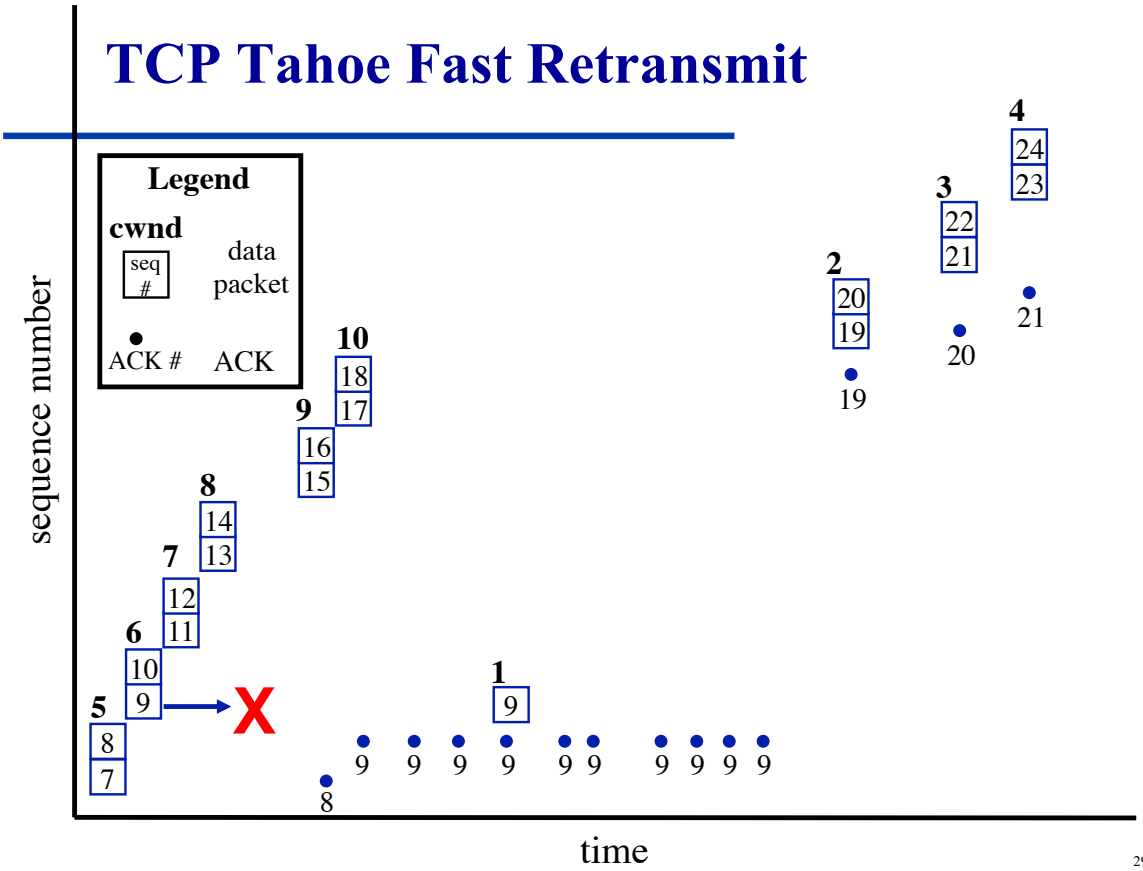
TCP Slow Start



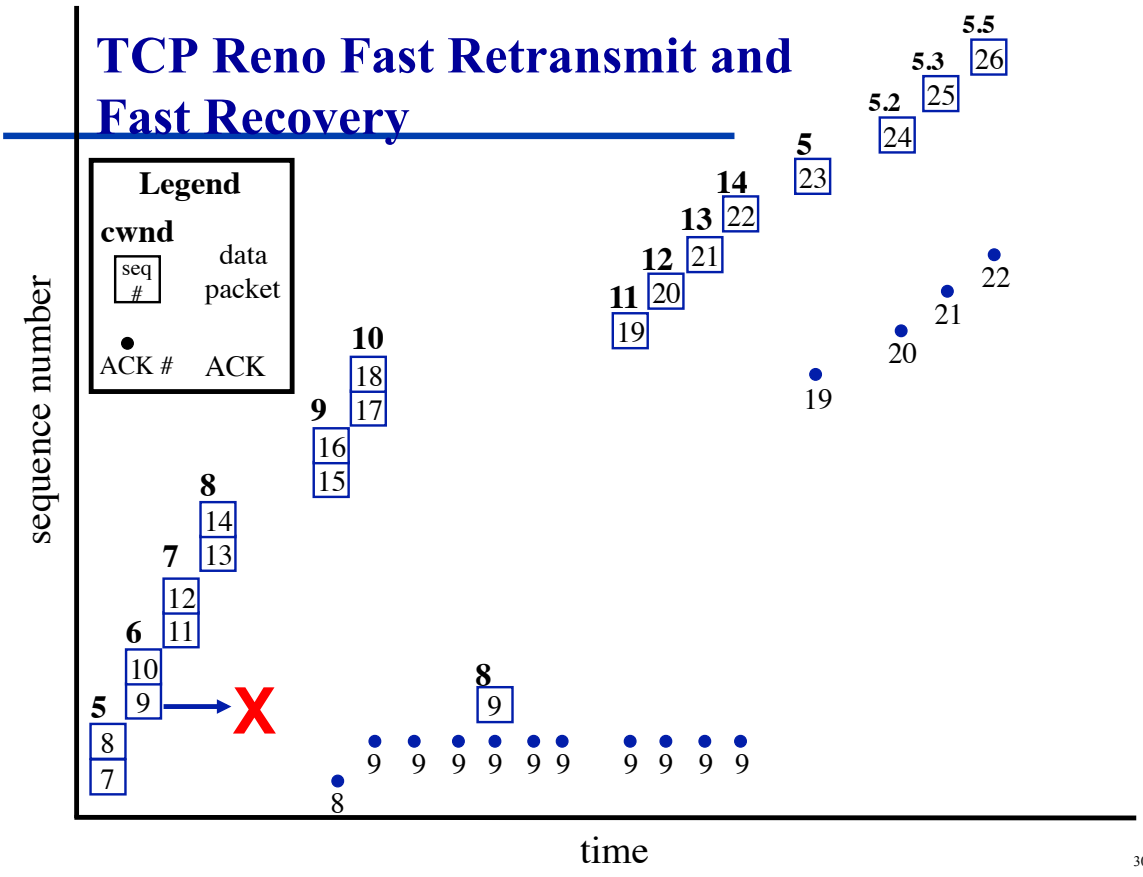
TCP Congestion Avoidance



TCP Tahoe Fast Retransmit



TCP Reno Fast Retransmit and Fast Recovery



Tahoe vs. Reno

One Lost Segment

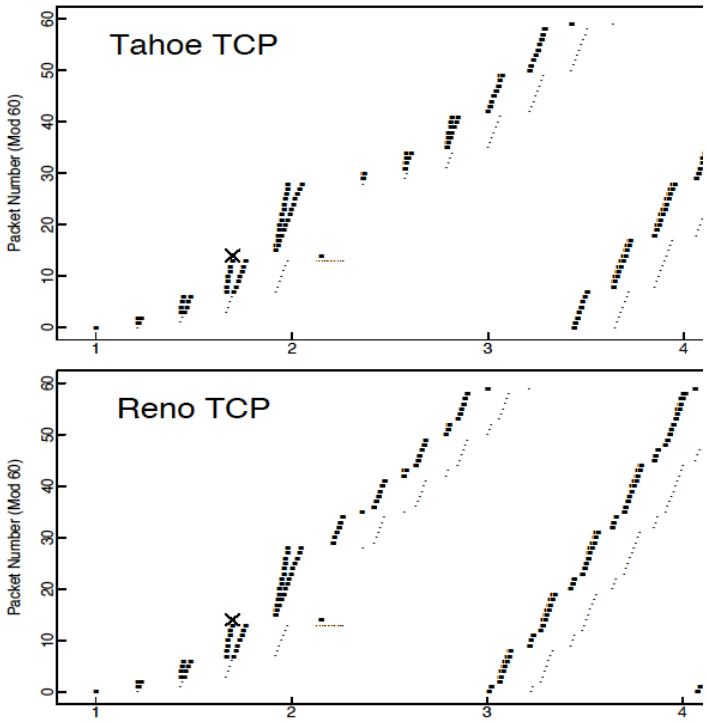


Figure 2 from "Simulation-based Comparison of Tahoe, Reno, and SACK TCP" by Fall and Floyd, SIGCOMM 1996.

Tahoe vs. Reno

Two Lost Segments

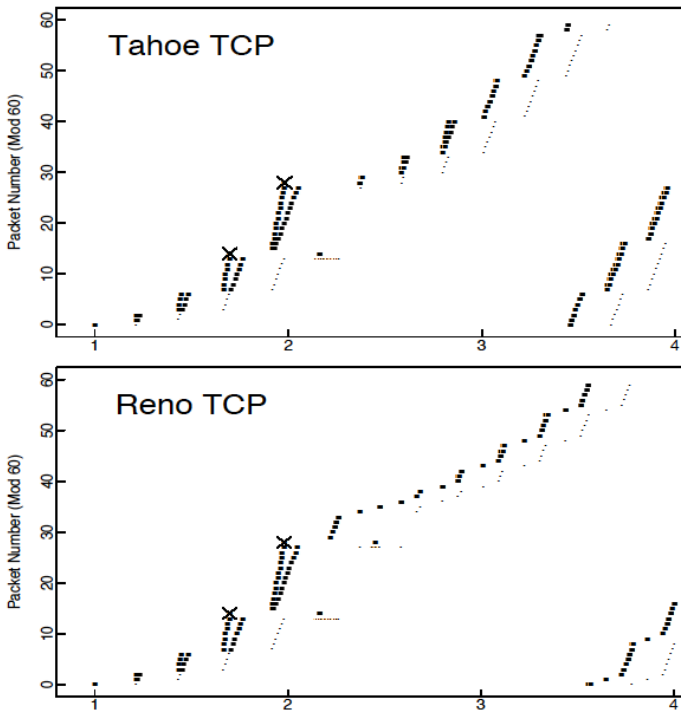


Figure 3 from "Simulation-based Comparison of Tahoe, Reno, and SACK TCP" by Fall and Floyd, SIGCOMM 1996.

Tahoe vs. Reno

Three Lost Segments

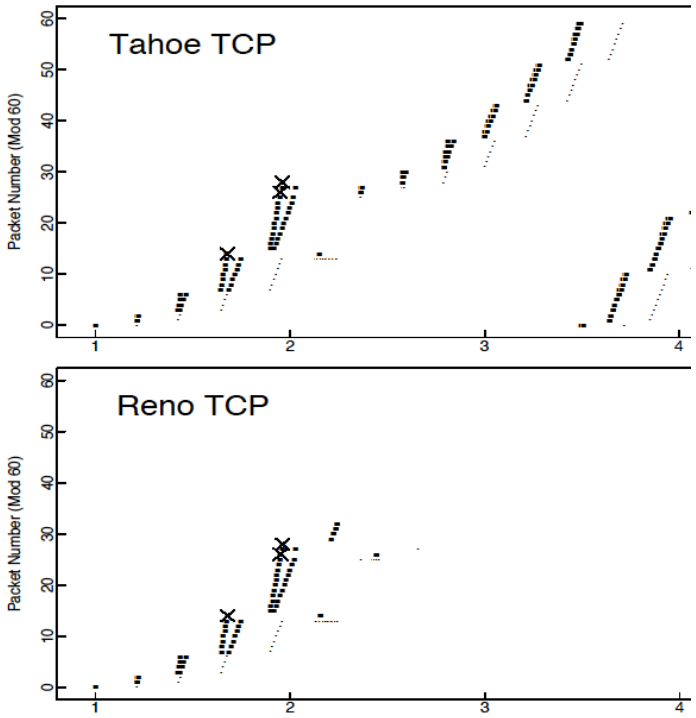


Figure 4 from "Simulation-based Comparison of Tahoe, Reno, and SACK TCP" by Fall and Floyd, SIGCOMM 1996.

NewReno

- TCP Reno
  - » fast recovery ends as soon as an ACK for the lost segment is received
  - » only one retransmission can be sent during each fast recovery period
- TCP NewReno
  - » *partial ACK* - acknowledges some, but not all, of the data sent before the segment loss was detected
  - » sender can infer that additional segments were lost
  - » allows sender to retransmit more than one segment during a single fast recovery
  - » only one lost segment may be retransmitted each RTT

Reno vs. NewReno

Two Lost Segments

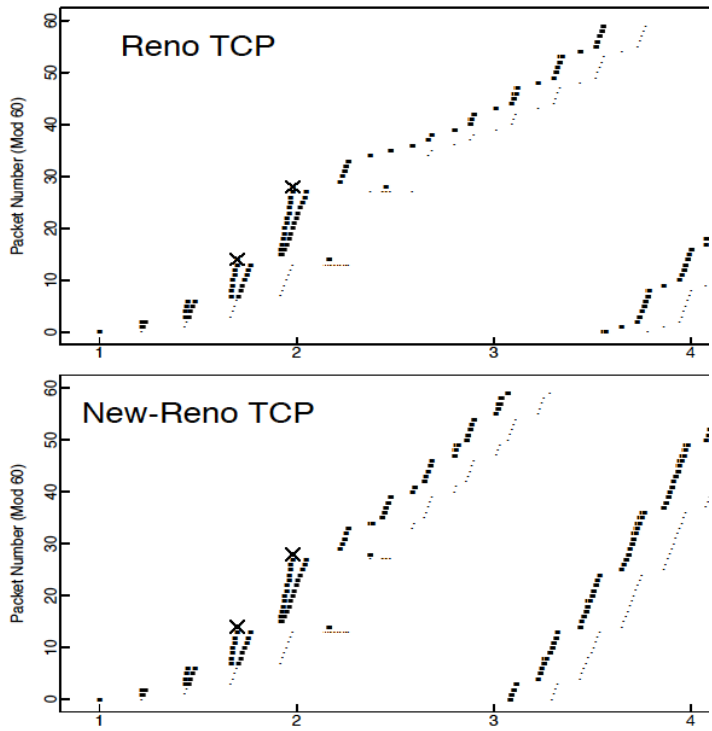


Figure 3 from "Simulation-based Comparison of Tahoe, Reno, and SACK TCP" by Fall and Floyd, SIGCOMM 1996.

Reno vs. NewReno

Three Lost Segments

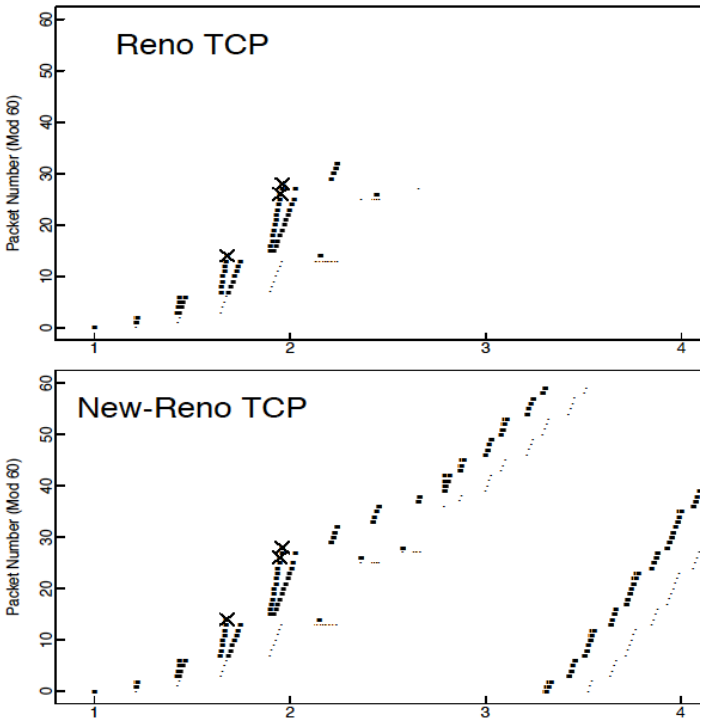
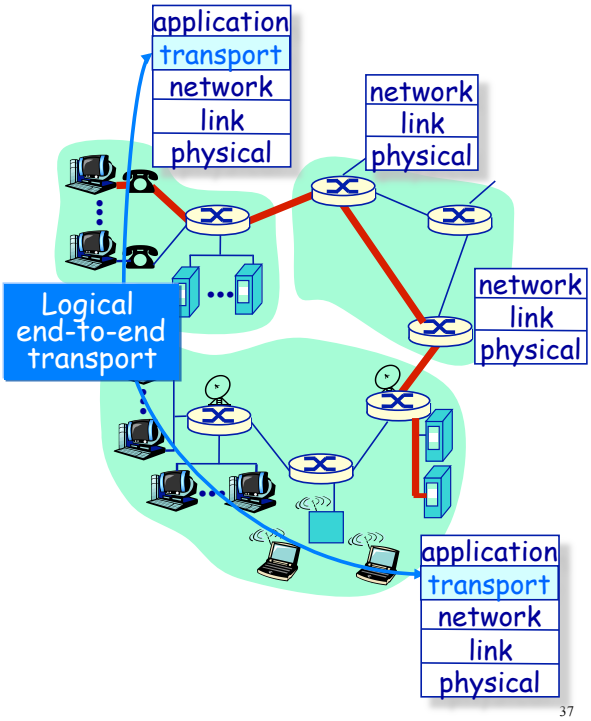


Figure 4 from "Simulation-based Comparison of Tahoe, Reno, and SACK TCP" by Fall and Floyd, SIGCOMM 1996.

Transport Layer Protocols & Services

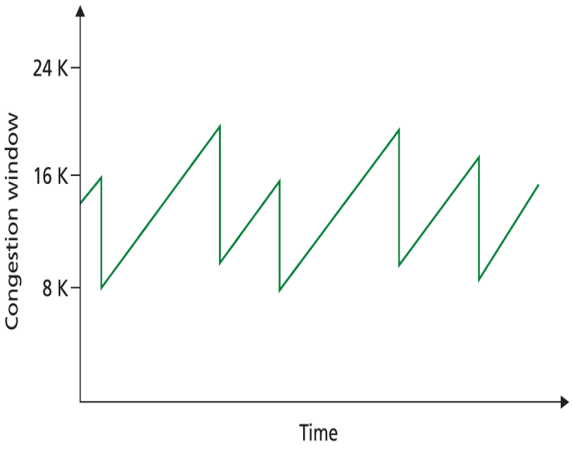
Performance issues

- What throughputs are attainable under TCP's congestion control scheme?
  - » What is the impact of slow-start/AIMD congestion control on throughput?
- How does congestion control impact the latency of TCP transfers?



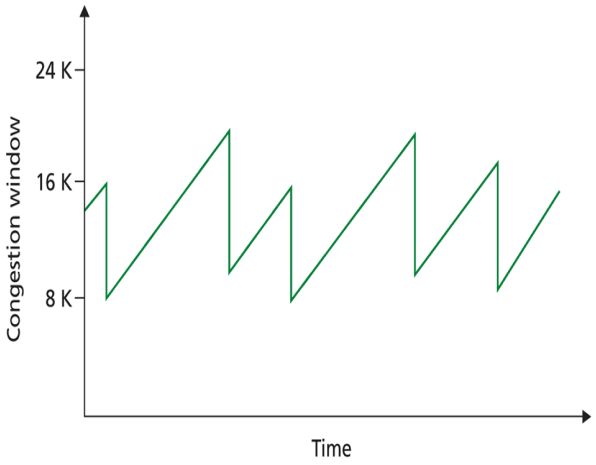
TCP Throughput

- TCP "sawtooth" Behavior
- What's *average* throughput for a long-lived connection?
  - » Ignore slow-start
- What's current rate?
  - » Current window size - *w*
  - » Current round-trip time - *RTT*
  - » *w/RTT*
- What if loss occurs?



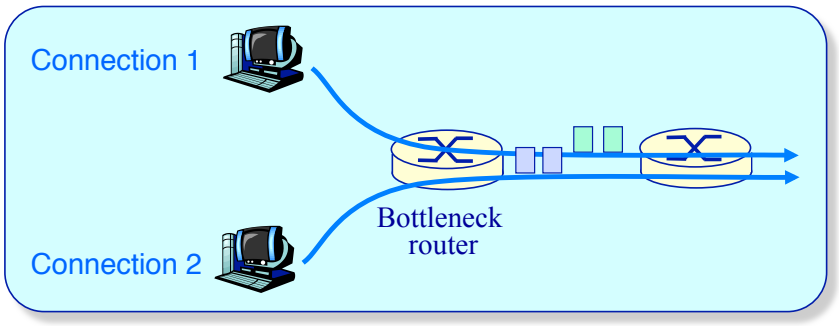
TCP Throughput

- *W* - window size when loss occurs
- Window size drops to *W/2*
  - » Rate - *W/2RTT*
- Assume *W* and *RTT* remain relatively constant
  - » New rate ranges from *W*/2*RTT* to *W/RTT*
  - » Increases by MSS/RTT every RTT
- Average throughput (*rate*)
  - » 0.75 *W/RTT*



TCP Performance

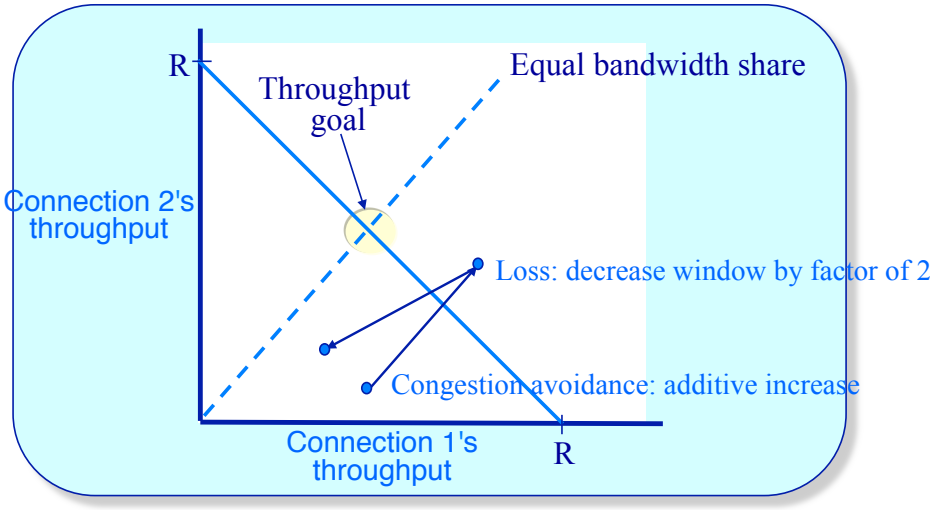
Is TCP throughput fairly realized?



- Simple fairness
  - » If *n* TCP sessions share a bottleneck link, each should get 1/*n* of link capacity
- MAX-MIN fairness
  - » If a connection receives less bandwidth than it requires then it receives the same amount of bandwidth as all other unsatisfied connection

TCP Throughput

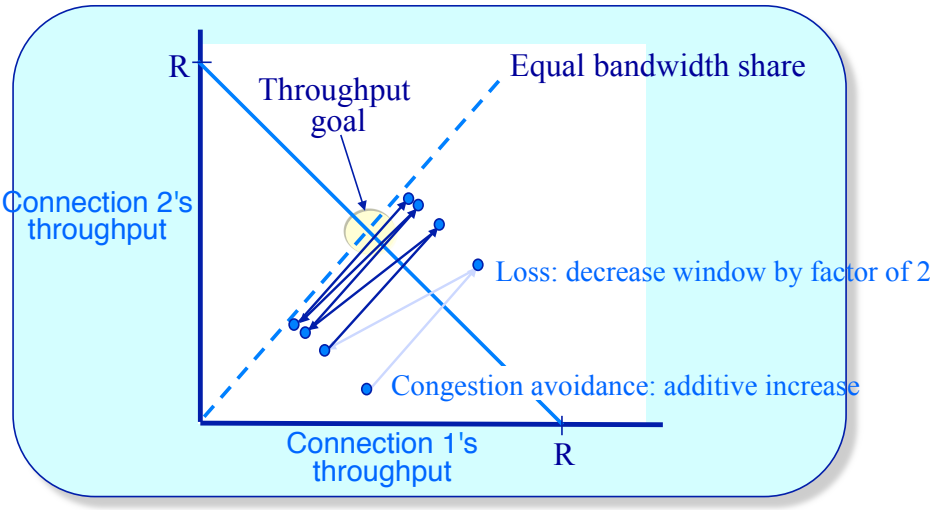
Is TCP fair?



- Consider two competing connections with same *MSS* and *RTT*
  - » Additive increase gives slope of 1, as throughput increases
  - » Multiplicative decrease decreases throughput proportionally

TCP Throughput

Is TCP fair?



- Consider two competing connections with same *MSS* and *RTT*
  - » Additive increase gives slope of 1, as throughput increases
  - » Multiplicative decrease decreases throughput proportionally

Fairness

UDP and Parallel TCP

UDP

- Multimedia apps often do not use TCP
  - » do not want rate throttled by congestion control
- Instead use UDP:
  - » pump audio/video at constant rate, tolerate packet loss
- Research area: TCP-friendly multimedia protocols

Parallel TCP connections

- Nothing prevents app from opening parallel connections between 2 hosts.
  - » web browsers do this
- Example: link of rate R supporting 9 existing connections
  - » new app asks for 1 TCP, gets rate R/10
  - » new app asks for 11 TCPs, gets R/2!

Transport Layer Protocols & Services

Summary

- Fundamental transport layer services
  - » Multiplexing/Demultiplexing
  - » Error detection
  - » Reliable data delivery
  - » Pipelining
  - » Flow control
  - » Congestion control
- Internet transport protocols
  - » UDP
  - » TCP