



15-441 Computer Networking

Lecture 19 – TCP Performance

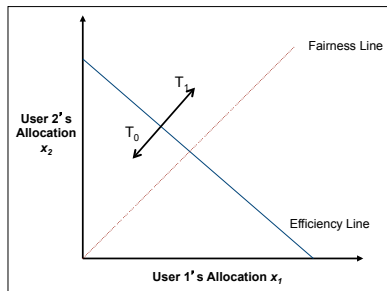
Outline

- TCP congestion avoidance
- TCP slow start
- TCP modeling

2

Additive Increase/Decrease

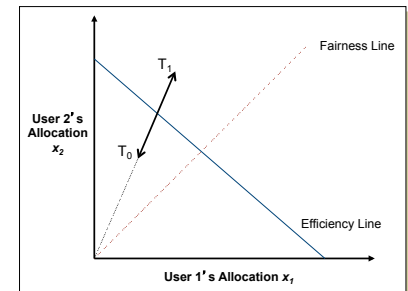
- Both X_1 and X_2 increase/ decrease by the same amount over time
 - Additive increase improves fairness and additive decrease reduces fairness



3

Multiplicative Increase/Decrease

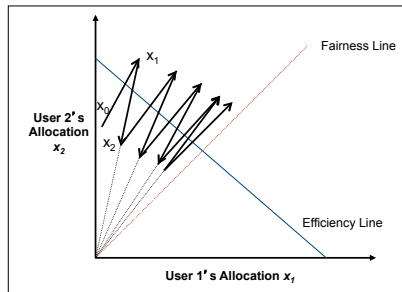
- Both X_1 and X_2 increase by the same factor over time
 - Extension from origin – constant fairness



4

What is the Right Choice?

- Constraints limit us to AIMD
 - Improves or keeps fairness constant at each step
 - AIMD moves towards optimal point



5

TCP Congestion Control

- Changes to TCP motivated by ARPANET congestion collapse
- Basic principles
 - AIMD
 - Packet conservation
 - Reaching steady state quickly
 - ACK clocking

6

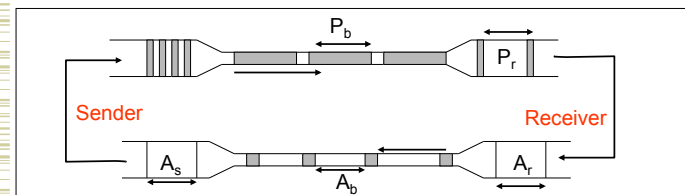
Implementation Issue

- Operating system timers are very coarse – how to pace packets out smoothly?
- Implemented using a congestion window that limits how much data can be in the network.
 - TCP also keeps track of how much data is in transit
- Data can only be sent when the amount of outstanding data is less than the congestion window.
 - The amount of outstanding data is increased on a “send” and decreased on “ack”
 - (last sent – last acked) < congestion window
- Window limited by both congestion and buffering
 - Sender's maximum window = Min (advertised window, cwnd)

7

ACK Clocking

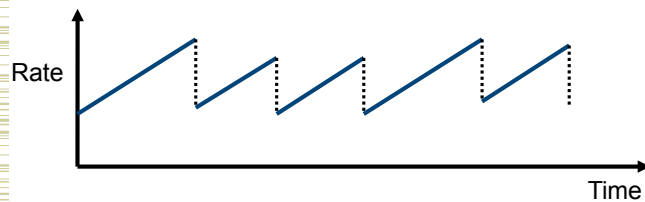
- Congestion window helps to “pace” the transmission of data packets
- In steady state, a packet is sent when an ack is received
 - Data transmission remains smooth, once it is smooth
 - Self-clocking behavior



8

AIMD

- Distributed, fair and efficient
- Packet loss is seen as sign of congestion and results in a multiplicative rate decrease
 - Factor of 2
- TCP periodically probes for available bandwidth by increasing its rate



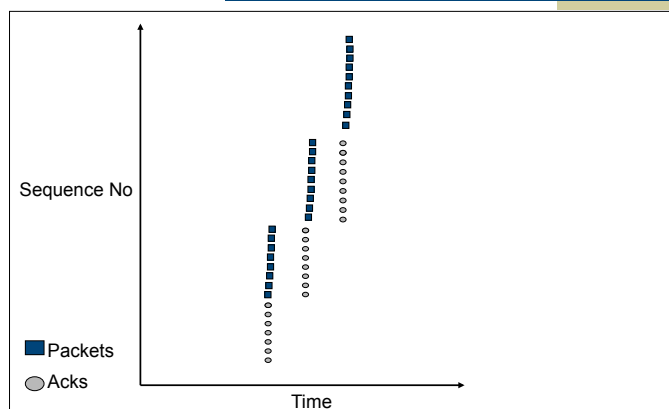
9

Congestion Avoidance

- If loss occurs when $cwnd = W$
 - Network can handle $0.5W \sim W$ segments
 - Set $cwnd$ to $0.5W$ (multiplicative decrease)
- Upon receiving ACK
 - Increase $cwnd$ by $(1 \text{ packet})/cwnd$
 - What is 1 packet? \rightarrow 1 MSS worth of bytes
 - After $cwnd$ packets have passed by \rightarrow approximately increase of 1 MSS
- Implements AIMD

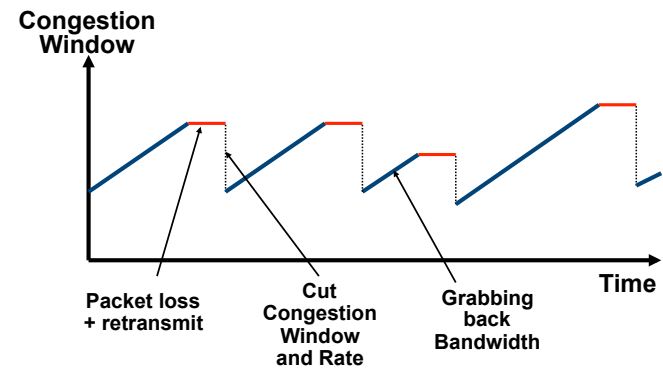
10

Congestion Avoidance Sequence Plot



11

Congestion Avoidance Behavior



12

Packet Conservation



- At equilibrium, inject packet into network only when one is removed
 - Sliding window and not rate controlled
 - But still need to avoid sending burst of packets → would overflow links
 - Need to carefully pace out packets
 - Helps provide stability
- Need to eliminate spurious retransmissions
 - Accurate RTO estimation
 - Better loss recovery techniques (e.g. fast retransmit)

13

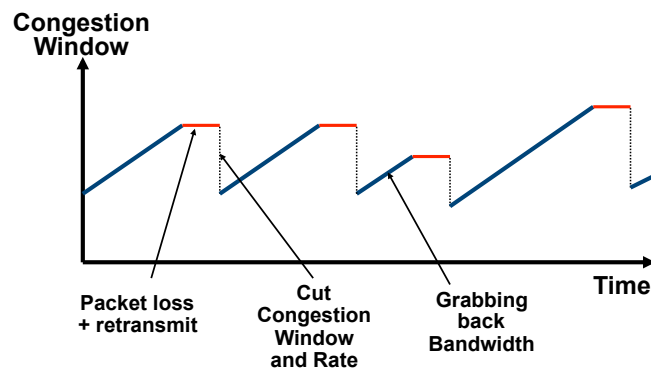
Outline



- TCP congestion avoidance
- **TCP slow start**
- TCP modeling

17

Congestion Avoidance Behavior



18

Reaching Steady State

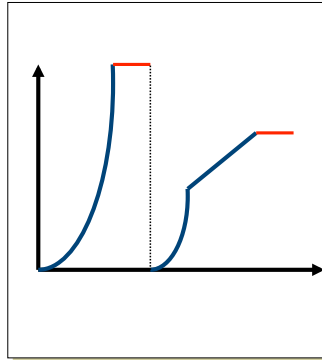


- Doing AIMD is fine in steady state but slow...
- How does TCP know what is a good initial rate to start with?
 - Should work both for a CDPD (10s of Kbps or less) and for supercomputer links (10 Gbps and growing)
- Quick initial phase to help get up to speed (slow start)

19

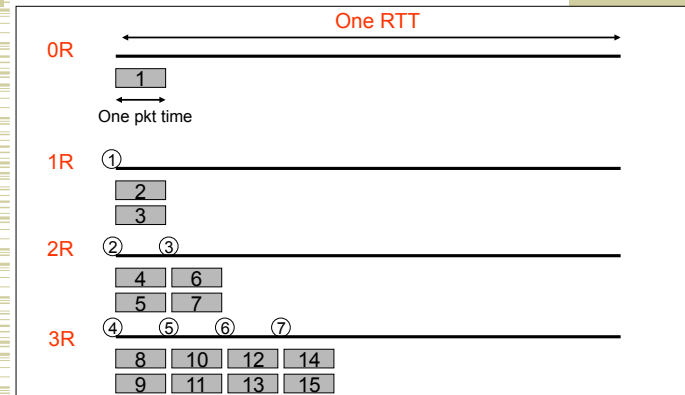
Slow Start Packet Pacing

- How do we get this clocking behavior to start?
 - Initialize $cwnd = 1$
 - Upon receipt of every ack, $cwnd = cwnd + 1$
- Implications
 - Window actually increases to W in $RTT * \log_2(W)$
 - Can overshoot window and cause packet loss



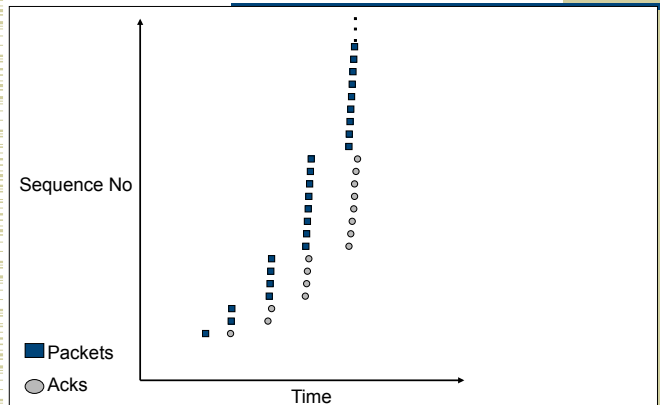
20

Slow Start Example



21

Slow Start Sequence Plot



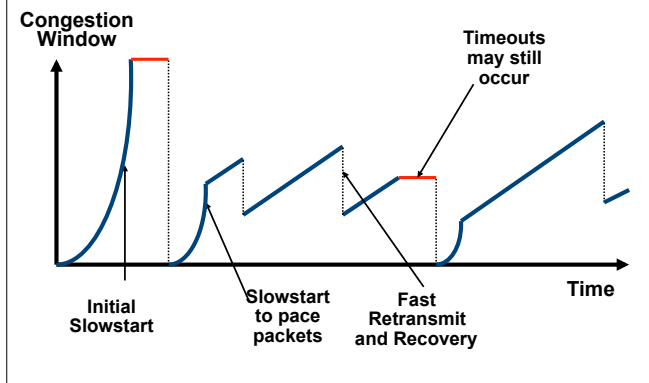
22

Return to Slow Start

- If packet is lost we lose our self clocking as well
 - Need to implement slow-start and congestion avoidance together
- When retransmission occurs set $ssthresh$ to $0.5w$
 - If $cwnd < ssthresh$, use slow start
 - Else use congestion avoidance

23

TCP Saw Tooth Behavior



24

Outline

- TCP congestion avoidance
- TCP slow start
- **TCP modeling**

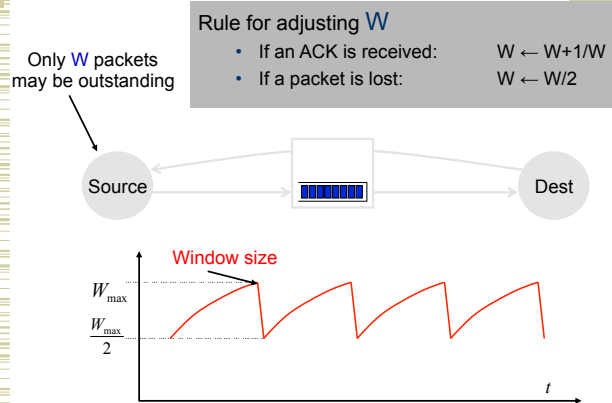
25

TCP Performance

- Can TCP saturate a link?
- Congestion control
 - Increase utilization until... link becomes congested
 - React by decreasing window by 50%
 - Window is proportional to rate * RTT
- Doesn't this mean that the network oscillates between 50 and 100% utilization?
 - Average utilization = 75%??
 - No...this is *not* right!

26

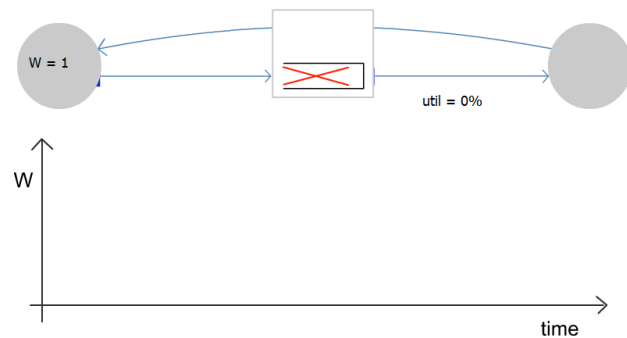
TCP Congestion Control



27

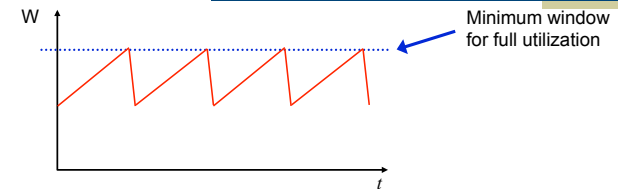
Single TCP Flow

Router *without* buffers



28

Summary Unbuffered Link



- The router can't fully utilize the link
 - If the window is too small, link is not full
 - If the link is full, next window increase causes drop
 - With no buffer it still achieves 75% utilization

29

TCP Performance

- In the real world, router queues play important role
 - Window is proportional to rate * RTT
 - But, RTT changes as well the window
 - Window to fill links = propagation RTT * bottleneck bandwidth
 - If window is larger, packets sit in queue on bottleneck link

30

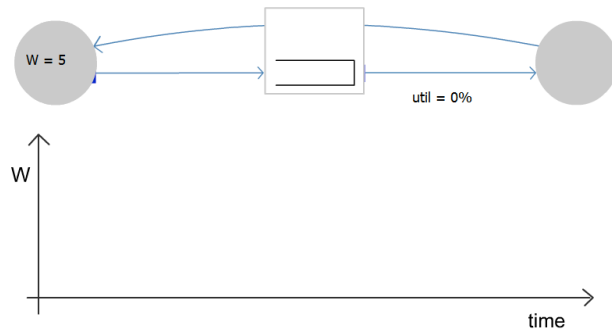
TCP Performance

- If we have a large router queue → can get 100% utilization
 - But, router queues can cause large delays
- How big does the queue need to be?
 - Windows vary from $W \rightarrow W/2$
 - Must make sure that link is always full
 - $W/2 > RTT * BW$
 - $W = RTT * BW + Qsize$
 - Therefore, $Qsize > RTT * BW$
 - Ensures 100% utilization
 - Delay?
 - Varies between RTT and $2 * RTT$

31

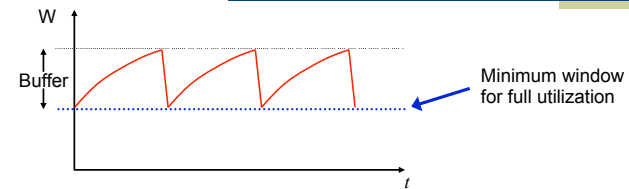
Single TCP Flow

Router with large enough buffers for full link utilization



32

Summary Buffered Link



- With sufficient buffering we achieve full link utilization
 - The window is always above the critical threshold
 - Buffer absorbs changes in window size
 - Buffer Size = Height of TCP Sawtooth
 - Minimum buffer size needed is $2T \cdot C$
 - This is the origin of the rule-of-thumb

33

TCP (Summary)

- General loss recovery
 - Stop and wait
 - Selective repeat
- TCP sliding window flow control
- TCP state machine
- TCP loss recovery
 - Timeout-based
 - RTT estimation
 - Fast retransmit
 - Selective acknowledgements

34

TCP (Summary)

- Congestion collapse
 - Definition & causes
- Congestion control
 - Why AIMD?
 - Slow start & congestion avoidance modes
 - ACK clocking
 - Packet conservation
- TCP performance modeling
 - How does TCP fully utilize a link?
 - Role of router buffers

35