CS 925

Lecture 11

TCP Congestion Control

Tuesday, February 27, 2024

Network Congestion Control

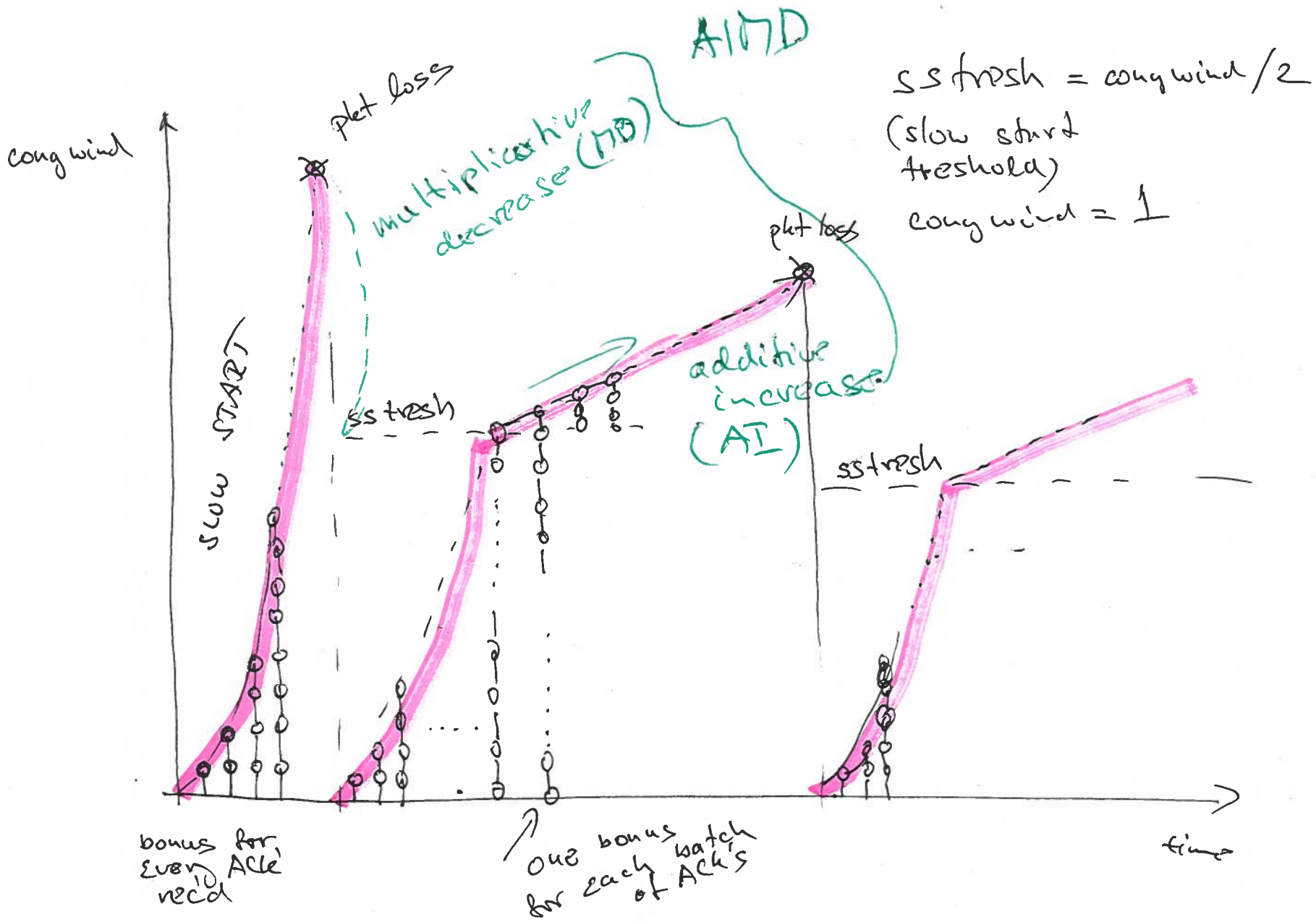
- **Method**:

  TransWind = min(*RecvWind*, *CongWind*)
  EffectiveWind = *TransWind* - (*LastByteSent* - *LastByteAckd*)

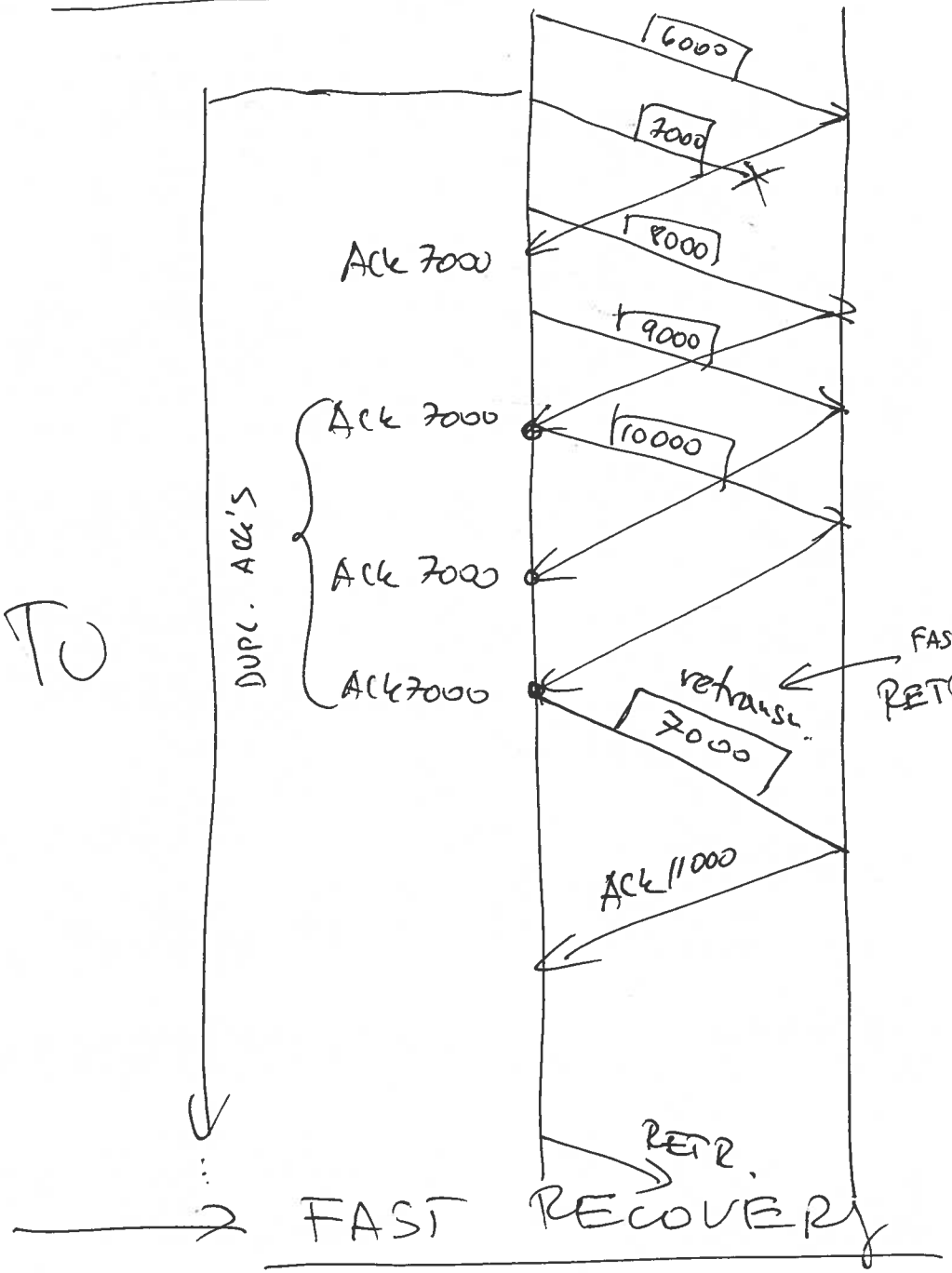
- *EffectiveWind* - used in transmission
- *RecvWind* - from Window Size field
- *CongWind* - transmitter's estimate of how many unacknowledged packets can be pushed onto the network without causing congestion

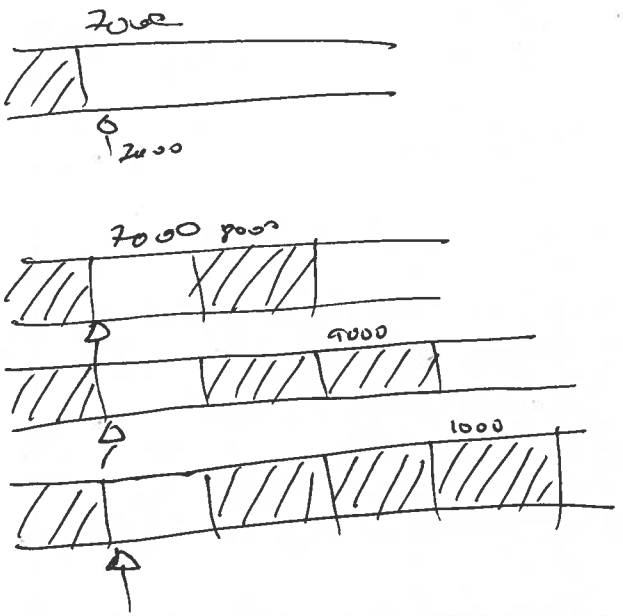
Congestion Window

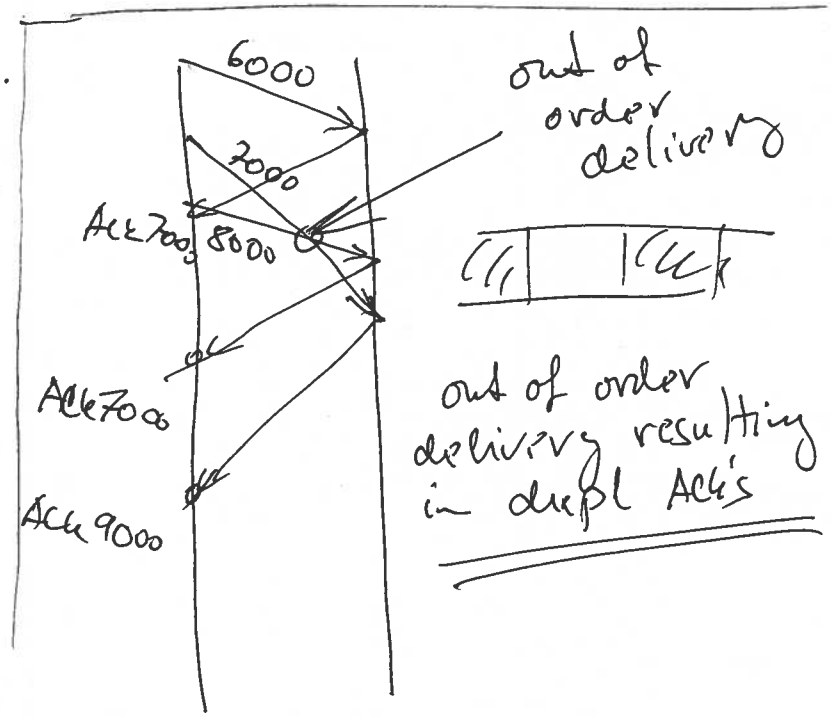
- Components algorithms of TCP network congestion control (RFC 2001):
  - Slow Start - initial growth of CongWind
  - Congestion Avoidance - AIMD-based “search” for optimal rate
  - Fast Retransmit - quick recovery from isolated packet losses
  - Fast Recovery - undoing congestion control steps under Fast Recovery


AIMD
ssthresh = congwind/2
(slow start threshold)
congwind = 1
congwind
pkt loss
multiplicative decrease (MD)
pkt loss
slow START
ssthresh
additive increase (AI)
ssthresh
bonus for every ACK needed
one bonus for each batch of ACK's
time

TCP FAST RETRANSM.







Variants of TCP (examples)

- Original TCP (RFC1122)
- TCP Tahoe (adds Fast Retransmit)
- TCP Reno (adds Fast Recovery)
- TCP Vegas (RTT-based)
- TCP BIC and CUBIC (Linux up to kernel 3.2)
- Compound TCP (Windows since Vista)
- TCP Proportional Rate Reduction (PRR) (Linux)
- TCP Bottleneck Bandwidth and Round-trip propagation time (BBR) (RTT-based, developed by Google)

TCP Vegas

- RTT observed
- An increase in RTT indicates congestion
  - reduce transmission rate
- Steady RTT measurements indicate underutilization
  - slowly increase transmission rate until RTT starts increasing