TCP - Summary

- Service
- Protocol Phases
- Error Control
- Congestion Control
- Flow Control

TCP Service:

TCP: Byte Stream → Ordered, reliable, well-paced

Protocol Phases

- SYN k → 3-way handshake
- SYN n; ACK k+1
- DATA k+1; ACK n+1
- ACK k+n+1
- data exchange
- FIN
- FIN ACK
-½ close
- Receiver window = \{P + 1, ..., P + N\}
- Where P = last received packet without gap
- When gets a packet in receiver window, ACK with sequence number of next expected packet
- Sender window = \{A + 1, ..., A + N\}
- Where A = last ACK received without gap
- Transmit packets in transmitter window
- If timeout for ACK(k), retransmit k, k+1, ...
- [Variation: fast retransmit: after 3DA]

Error Control: Go Back N

- Sender:
  - Transmitter window = \{A + 1, ..., A + N\}
  - Where A = last ACK received without gap
  - Transmit packets in transmitter window
  - If timeout for ACK(k), retransmit k, k+1, ...
  - [Variation: fast retransmit: after 3DA]
- Receiver:
  - Receiver window = \{P + 1, ..., P + N\}
  - Where P = last received packet without gap
  - When gets a packet in receiver window, ACK with sequence number of next expected packet

Congestion Control:

- Flows share links:
- How to share the links bandwidth?

Congestion Control

- What is fair sharing?
- Single bottleneck: Equal rates
- Multiple Bottlenecks?
  - Max Min?
  - More generally:

Maximize $\sum_i U_i(x_i)$
subject to $Ax \leq c$
TCP Algorithm: AIMD

- Slow Start: Start with $W = 1$
  - Discover quickly available throughput
  - Increase window fast
  - When TO: set $ssthresh = W/2$; restart SS until $ssthresh$; CA

- Congestion Avoidance: AIMD
  - $W = W + 1/W$ at each ACK (linear over time)
  - Fast retransmit + fast recovery
  - After timeout: $ssthresh = W/2$ → SS until $ssthresh$

- Fast Retransmit: $3DA$
  - $A = average, D = deviation$
  - Ignore retransmissions in calculations
  - After TO: Double timeout value; reset after new ACK

Refinements: Summary

- Flow Control
  - Objective: Avoid saturating destination
  - Algorithm: Receiver advertizes window RAW
    - $\text{window} = \min\{\text{RAW} - \text{OUT}, W\}$
    - $\text{OUT} = \text{Oustanding} = \text{Last sent} - \text{last ACKed}$
    - $W = \text{Cong. Window from AIMD + refinements}$

Flow Control

- Window = min(RAW – OUT, W)

Congestion Control: Summary

- Slow Start: Discover available bandwidth
- Congestion Avoidance: AIMD → Tries to be fair
- Refinements:
  - Fast Retransmit: $3DA$
  - Fast Recovery: Reset $W$ to $W/2$ (instead of $W = 1$)
    - More precisely: $ssthresh = W/2, W = ssthresh + 3, W = W + 1$ per DA after $3^{rd}$ DA, $W = ssthresh$ when get new ACK
  - TO: set $ssthresh = W/2, W = 1$, SS until $W = ssthresh$, then CA
- Timers:
  - Timeout = Average + 4 Deviations
  - If timeout → Timeout x 2
  - Reset after new packet or new ACK
- Flow Control:
  - Window = min(RAW – OUT, W)