CSE/EE 461 – Lecture 17
TCP Congestion Control

Janet Davis
jlnd@cs.washington.edu
February 13, 2004
Reading: Peterson 6.2.1; 6.3

Last time…
• Bandwidth Allocation (Ch. 6 in Peterson)
  – How can many senders share bandwidth across the Internet?
• Congestion & the bottleneck link
• Fairness & flows
• Taxonomy of bandwidth allocation strategies

Questions from last time…
• Still confused about taxonomy
• Rate-based vs window based?
• Can we combine the strategies? Are other points in the design space possible?
• How do routers and hosts implement bandwidth allocation?
• Can a router direct a packet to a longer route to avoid congestion?

This time…
• What’s in a router?
  – Scheduling and Buffer Management
• TCP Congestion Control
  – Additive Increase/Multiplicative Decrease (AIMD)
  – Slow Start
  – Fast Retransmit & Fast Recovery
What's in a Router?

- From routers or hosts
- “Router” routing forwarding
- To routers or hosts

Model of a Router

- Input Ports
  - Data Link and PHY
  - Queue
  - Switching Fabric
  - Queue
  - Forwarding this side
  - Processor
  - Queue
  - Scheduling and Buffering this side

- Output Ports
  - Data Link and PHY
  - Queue

Queue: FIFO with Tail Drop

- Arriving packets
- Need free buffers
- Need to transmit
- Free buffers
- Queued packets
- Drop

Scheduling and Buffer Management

- Two different functions implemented at the output queue
- A scheduling discipline
  - Order in which to send packets
  - e.g.,
- A buffer management policy
  - Which packets get dropped
  - e.g.,
TCP Before Congestion Control

- Number of packets in flight determined only by advertised flow control window
- Congestion collapse
  - Too many packets in flight
  - Timeouts & retransmission
  - Full buffers
  - Large delays

TCP Congestion Control

- Host-centric, feedback-based, window-based
- Idea: Dynamically vary the size of the sliding window to match available bandwidth
  - cwnd is the congestion window
  - Sliding window uses
- How do we decide what size cwnd should be?

TCP Probes the Network

- Each source independently probes the network to determine how much bandwidth is available
- Infer congestion from packet loss
  - Requires no support from routers
  - What does this assume?

TCP is “Self-Clocking”

- ACK is a signal that a packet has left the network and it’s safe to send another
- ACKs pace transmissions at approximately the bottleneck rate
**AIMD (Additive Increase/Multiplicative Decrease)**

- Increase slowly while we believe there is bandwidth
  - Additive increase
  - cwnd +=

- Decrease quickly when there is loss (went too far!)
  - Multiplicative decrease
  - cwnd /=

**TCP Sawtooth Pattern**

![TCP Sawtooth Pattern](image)

**“Slow Start”**

- What is the ideal value of cwnd? How long will AIMD take to get there?

- Use a different strategy to get close to ideal value
  - Double cwnd every RTT
  - Or, cwnd +=

**Combining Slow Start and AIMD**

- Use slow start whenever the connection has no packets in flight
  -
  -

- But we don’t want to overshoot the ideal cwnd, so remember the last cwnd that worked
  - Sthresh = cwnd after cwnd /= 2 on loss
  - Switch to AIMD once cwnd passes ssthresh after timeout
**Example (Slow Start + AIMD)**

![Graph showing data transfer over time for Slow Start + AIMD](image)

**Fast Retransmit**

- Cumulative ACKs => Duplicate ACKs after loss
- Lets us infer packet loss instead of waiting for a timeout
- TCP uses 3 duplicate ACKs to detect loss

**Fast Recovery**

- After Fast Retransmit, use further duplicate ACKs to grow cwnd and clock out new packets.
  - These ACKs represent packets that have left the network; it should be safe to send more.
- End result: Can use AIMD when there are single packet losses. Only slow start at the start of the connection.

**Example (with Fast Retransmit)**

![Graph showing data transfer over time for Fast Retransmit](image)
Example (with Fast Recovery)

Key Concepts

- Scheduling and buffer management
- TCP probes the network for bandwidth by using loss to infer congestion
- The congestion window is managed to be additive increase/multiplicative decrease
  - It took fast retransmit and fast recovery to get there
- Slow start is used to avoid lengthy initial delays
  - Ramp up to near target rate and then switch to AIMD
- Next time (Wednesday!): Congestion Avoidance