Lecture 7: Flow Control

CSE 123: Computer Networks
Alex C. Snoeren

No class Monday
Lecture 7 Overview

- Flow control
  - Go-back-N
  - Sliding window
Stop-and-Wait Performance

- Lousy performance if xmit 1 pkt $\ll$ prop. delay
  - How bad?

- Want to utilize all available bandwidth
  - Need to keep more data “in flight”
  - How much? Remember the bandwidth-delay product?

- Also limited by quality of timeout (how long?)
Pipelined Transmission

- Keep multiple packets “in flight”
  - Allows sender to make efficient use of the link
  - Sequence numbers ensure receiver can distinguish frames

- Duplicate acknowledgements signal loss
  - ACK the highest *consecutive* frame received
  - Ignore (for now) non-sequential frames

CSE 123 – Lecture 7: Flow Control
Go-Back-

- Retransmit from point of loss upon duplicate ACK
  - Packets between loss event and retransmission are ignored
  - Also “go-back-N” if a timeout event occurs
- ACKs are cumulative
  - Acknowledge current frame and all previous ones
Send Window

- Bound on number of outstanding packets
  - Window “opens” upon receipt of new ACK
  - Window resets entirely upon a timeout

- Limits amount of waste
  - Still lots of duplicates
  - We can do better with selective retransmission

Go-Back-N Example with window size 3
Sliding Window

- Single mechanism that supports:
  - Multiple outstanding packets
  - Reliable delivery
  - In-order delivery
  - Flow control

- At the core of all modern ARQ protocols

- Stop-and-Wait is a special case
  - Receive window size of one
Window bounds outstanding unACKed data
  - Implies need for buffering at sender
“Last” ACK applies to in-order data
What to do on a timeout?
  - Go-Back-N: send all unacknowledged data on timeout
  - Selective Repeat: timer per packet, resend as needed
Sliding Window – Receiver

Receiver buffers too:
- data may arrive out-of-order
- or faster than can be consumed—flow control

Receiver ACK choices:
- Cumulative, Selective (exempt missing frames), Negative
Deciding When to Retransmit

- How do you know when a packet has been lost?
  - Ultimately sender uses timers to decide when to retransmit

- But how long should the timer be?
  - Too long: inefficient (large delays, poor use of bandwidth)
  - Too short: may retransmit unnecessarily (causing extra traffic)

- Right timer is based on the round-trip time (RTT)
  - Which can vary greatly for reasons well see later
Can we shortcut the timeout?

- Timeout is long in practice

- If packets are usually in order then out-of-order ACKs imply that a packet was lost
  - Negative ACK
    » Receiver requests missing packet
  - Fast retransmit
    » When sender receives multiple duplicate acknowledgements resends missing packet
Fast retransmit

- Don’t bother waiting
  - Receipt of duplicate acknowledgement (dupACK) indicates loss
  - Retransmit immediately

- Used in TCP
  - Need to be careful if frames can be reordered
Is ARQ the Only Way?

- No. We could use redundancy
  - Send additional data to compensate for lost packets

- Why not use retransmission?
  - Broadcast media with lots of receivers
    - If each one ACK/NAK then hard to scale
      - Lots of messages
      - Lots of state
    - Heterogeneous receivers
      - E.g., variable quality wireless reception
  - Highly lossy or very long delay channels (e.g., satellite)
For Next Time

- Keep reading 2.6 in P&D
- Keep going on the project…
- NO CLASS MONDAY!