Reliable Transmission: Pipelined Transfer

Srinidhi Varadarajan

Pipelined protocols

Pipelining: sender allows multiple, “in-flight”, yet-to-be-acknowledged pkts
- range of sequence numbers must be increased
- buffering at sender and/or receiver

Two generic forms of pipelined protocols: go-Back-N, selective repeat

---

Go-Back-N

Sender:
- k-bit seq # in pkt header
- “window” of up to N, consecutive unack’ed pkts allowed

\[ \text{send base, not seq num} \]

\[ \text{sent, not yet ack'd} \]

\[ \text{not usable} \]

- ACK(n): ACKs all pkts up to, including seq # n - “cumulative ACK”
  - may receive duplicate ACKs (see receiver)
- timer for each in-flight pkt
- timeout(n): retransmit pkt n and all higher seq # pkts in window

GBN: sender extended FSM

GBN in action

---

GBN: receiver extended FSM

receiver simple:
- ACK-only: always send ACK for correctly-received pkt with highest in-order seq #
  - may generate duplicate ACKs
  - need only remember expected seq num
- out-of-order pkt:
  - discard (don’t buffer) \( \rightarrow \) no receiver buffering!
  - ACK pkt with highest in-order seq #
Problems with GBN

- Retransmits entire sender window on timeout
  - Can cause excessive retransmissions
  - Problem is exacerbated for networks with large "memory", i.e. large delay bandwidth product
- Receiver throws away any out of order packets, even if they are received correctly.
  - Forces retransmission

Selective Repeat

- receiver individually acknowledges all correctly received pkts
  - buffers pkts, as needed, for eventual in-order delivery to upper layer
- sender only resends pkts for which ACK not received
  - sender timer for each unACKed pkt
- sender window
  - N consecutive seq #’s
  - again limits seq #’s of sent, unACKed pkts

Selective repeat: sender, receiver windows

Selective repeat

- receiver
  - data from above:
    - if next available seq # in window, send pkt
    - else hold packet
  - timeout(n):
    - resend pkt n, restart timer
  - ACK(n) in [sendbase, sendbase+N):
    - mark pkt n as received
    - if n smallest unACKed pkt, advance window base to next unACKed seq
    - Transmit any pending packets
  - pkt n in [rcvbase, rcvbase+N-1]:
    - send ACK(n)
    - out-of-order: buffer
    - in-order: deliver (also deliver buffered, in-order pkts), advance window to next not-yet-received pkt
  - pkt n in [rcvbase-N,rcvbase-1]:
    - ACK(n)
  - otherwise:
    - ignore
Selective repeat: dilemma

Example:
- seq #s: 0, 1, 2, 3
- window size=3
- receiver sees no difference in two scenarios!
- incorrectly passes duplicate data as new in (a)

Q: what relationship between seq # size and window size?

Out of Order Delivery

- What happens if the network delivers packets out of order
  - Send order ≠ receive order
- Need a much larger sequence space
  - Why?
- TCP faces the same problem. IP does not guarantee in-order packet delivery
  - TCP assumes max. packet lifetime of 3 minutes. Seq. numbers should be reused within 3 minutes. RFC 1323

Reading

Required