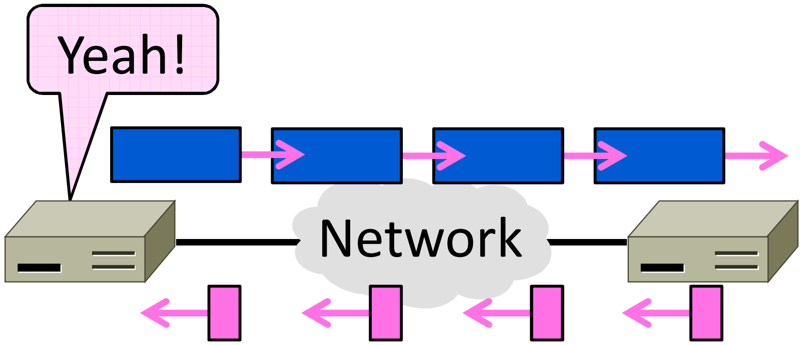
Introduction to Computer Networks

Sliding Windows (§3.4, §6.5.8)

David Wetherall (djw@uw.edu)
Professor of Computer Science & Engineering
UNIVERSITY of WASHINGTON

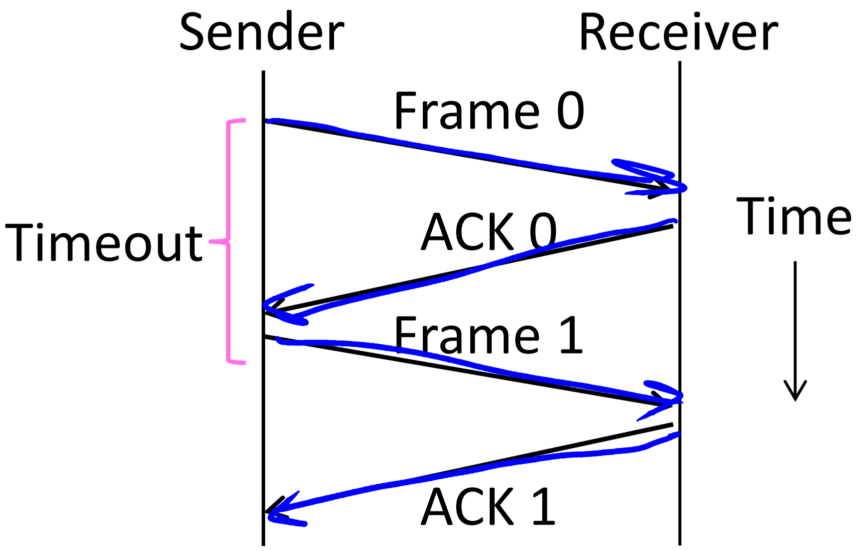
Topic

- The sliding window algorithm
  - Pipelining and reliability
  - Building on Stop-and-Wait



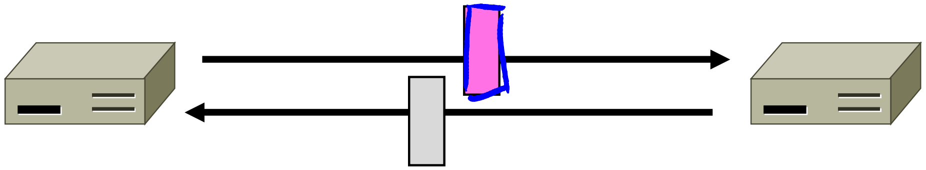
Recall

- ARQ with one message at a time is Stop-and-Wait (normal case below)



Limitation of Stop-and-Wait

- It allows only a single message to be outstanding from the sender:
  - Fine for LAN (only one frame fit)
  - Not efficient for network paths with BD >> 1 packet



Limitation of Stop-and-Wait (2)

- Example: R=1 Mbps, D = 50 ms
  - RTT (Round Trip Time) = 2D = 100 ms
  - How many packets/sec?

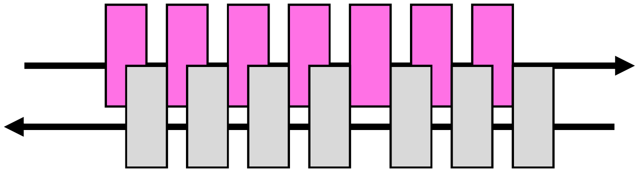
    10 packets/sec = 100 kbps
    ~10%

  - What if R=10 Mbps?

    10 packets/sec = 100 kbps ~1%

Sliding Window

- Generalization of stop-and-wait
  - Allows W packets to be outstanding
  - Can send W packets per RTT (=2D)



  - Pipelining improves performance
  - Need W=2BD to fill network path

Sliding Window (2)

- What W will use the network capacity?
- Ex: R=1 Mbps, D = 50 ms

$W = 2BD = 10^{6} \cdot 100 \cdot 10^{3}$ 100 kbit

$\sim$ 10 packets

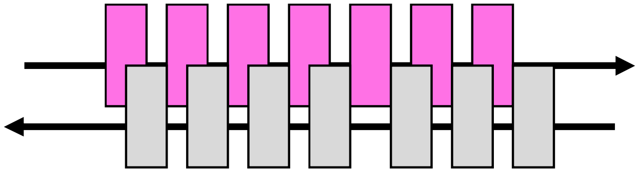
- Ex: What if R=10 Mbps?

$W = 2BD = 1000$ kbit

$\sim$ 100 packets

Sliding Window (3)

- Ex: R=1 Mbps, D = 50 ms
  - 2BD = 10^6 b/sec x 100. 10^{-3} sec = 100 kbit
  - W = 2BD = 10 packets of 1250 bytes



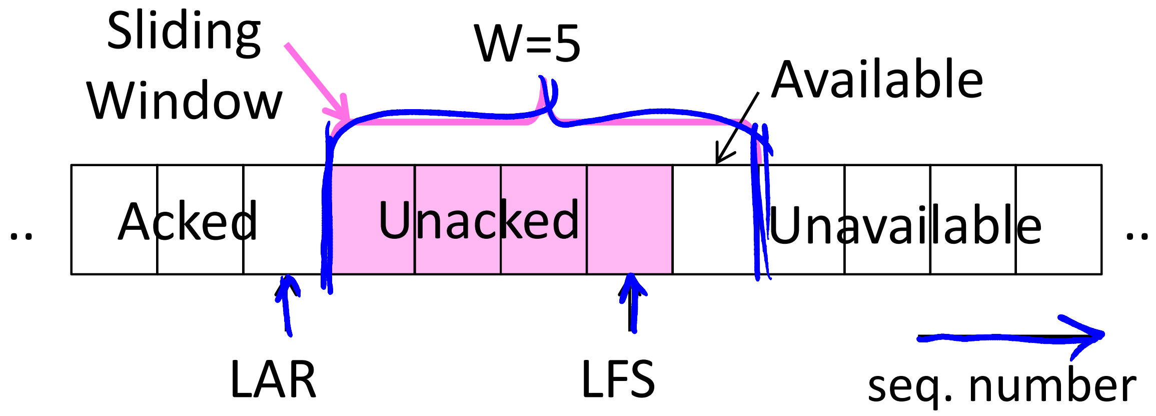
- Ex: What if R=10 Mbps?
  - 2BD = 1000 kbit
  - W = 2BD = 100 packets of 1250 bytes

Sliding Window Protocol

- Many variations, depending on how buffers, acknowledgements, and retransmissions are handled
- Go-Back-N »
  - Simplest version, can be inefficient
- Selective Repeat »
  - More complex, better performance

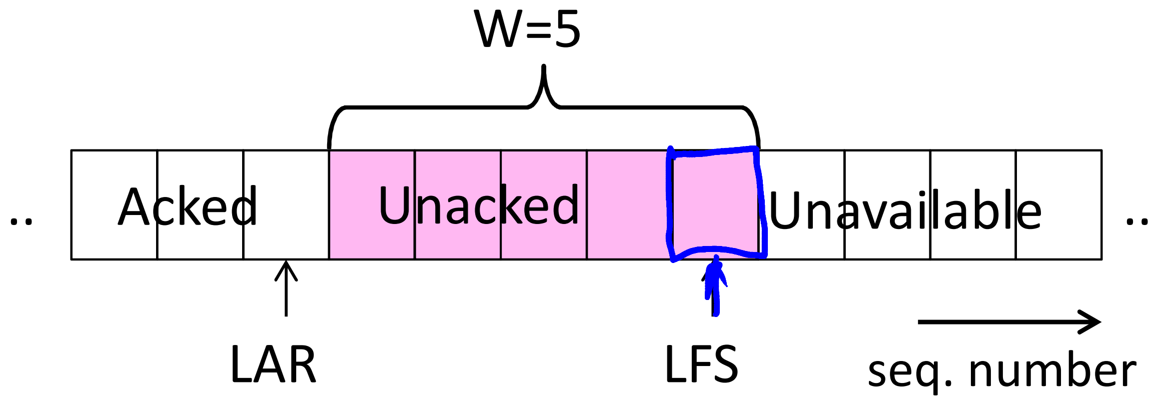
Sliding Window – Sender

- Sender buffers up to W segments until they are acknowledged
  - LFS=LAST FRAME SENT, LAR=LAST ACK REC'D
  - Sends while LFS – LAR ≤ W



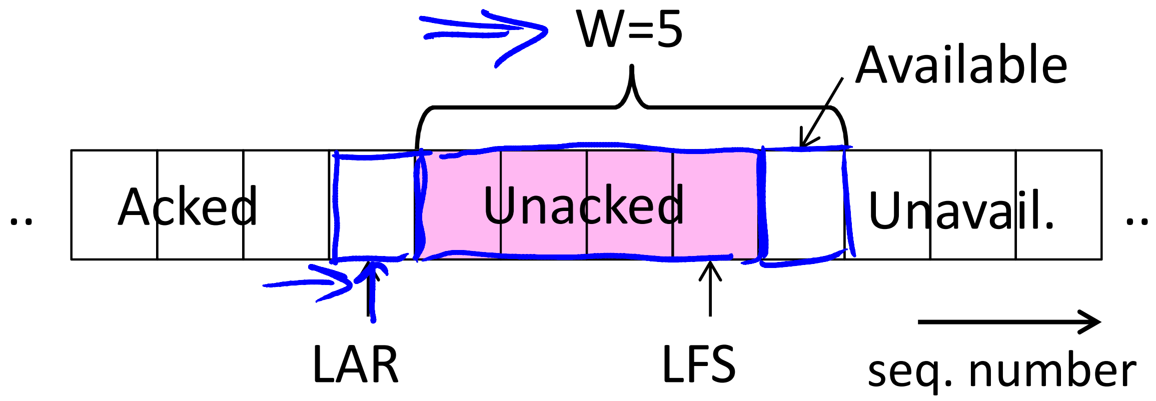
Sliding Window – Sender (2)

- Transport accepts another segment of data from the Application ...
  - Transport sends it (as LFS–LAR → 5)



Sliding Window – Sender (3)

- Next higher ACK arrives from peer…
  - Window advances, buffer is freed
  - LFS–LAR → 4 (can send one more)



Sliding Window – Go-Back-N

- Receiver keeps only a single packet buffer for the next segment
  - State variable, LAS = LAST ACK SENT

- On receive:
  - If seq. number is LAS+1, accept and pass it to app, update LAS, send ACK
  - Otherwise discard (as out of order)

Sliding Window – Selective Repeat

- Receiver passes data to app in order, and buffers out-of-order segments to reduce retransmissions

- ACK conveys highest in-order segment, plus hints about out-of-order segments

- TCP uses a selective repeat design; we'll see the details later

Sliding Window – Selective Repeat (2)

- Buffers W segments, keeps state variable, LAS = LAST ACK SENT

- On receive:
  - Buffer segments [LAS+1, LAS+W]
  - Pass up to app in-order segments from LAS+1, and update LAS
  - Send ACK for LAS regardless

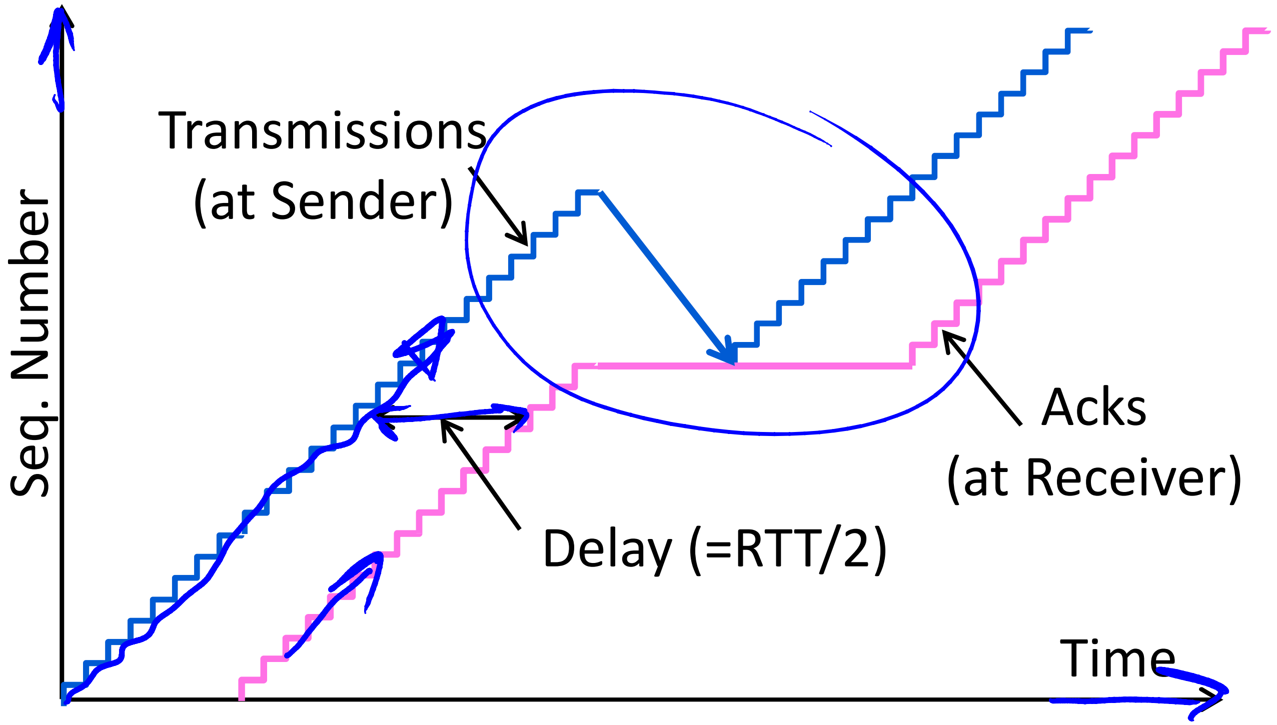
Sliding Window – Retransmissions

- Go-Back-N sender uses a single timer to detect losses
  - On timeout, resends buffered packets starting at LAR+1
- Selective Repeat sender uses a timer per unacked segment to detect losses
  - On timeout for segment, resend it
  - Hope to resend fewer segments

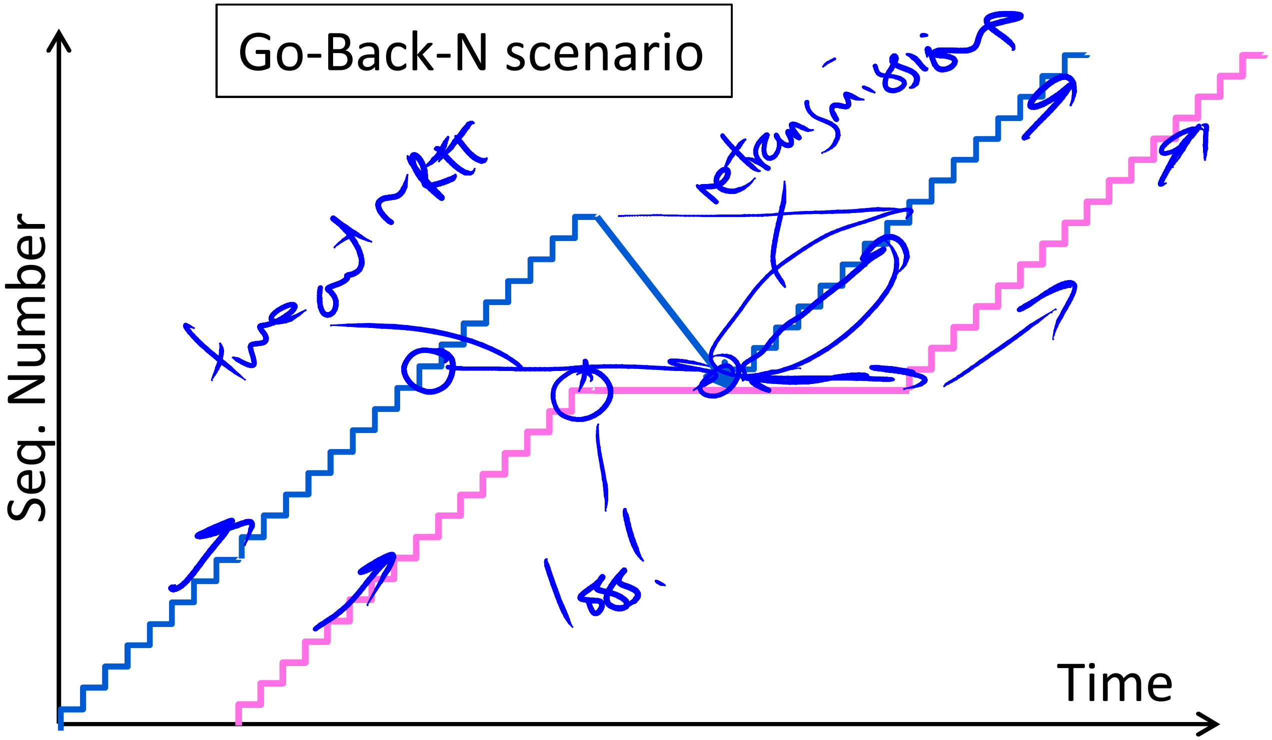
Sequence Numbers

- Need more than 0/1 for Stop-and-Wait …
  - But how many?
- For Selective Repeat, need W numbers for packets, plus W for acks of earlier packets
  - 2W seq. numbers
  - Fewer for Go-Back-N (W+1)
- Typically implement seq. number with an N-bit counter that wraps around at 2^N—1
  - E.g., N=8: …, 253, 254, 255, 0, 1, 2, 3, …

Sequence Time Plot

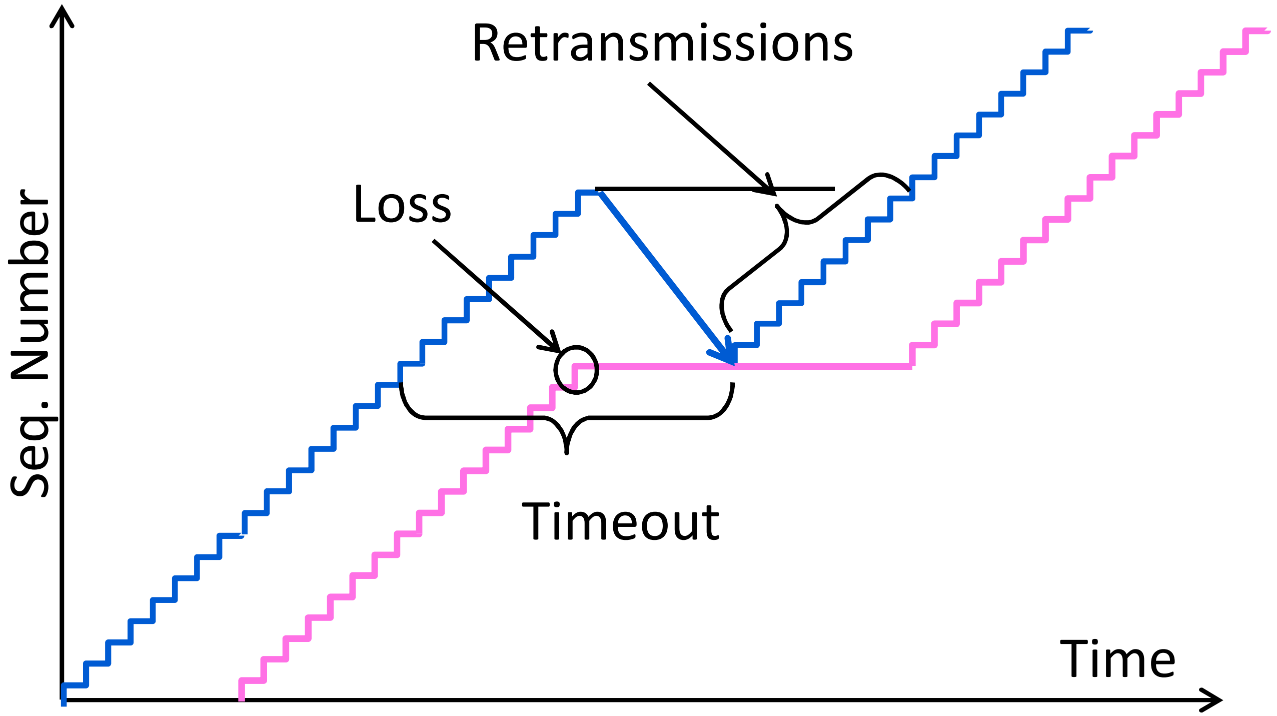


Sequence Time Plot (2)



Go-Back-N scenario

Sequence Time Plot (3)



END

Slide material from: TANENBAUM, ANDREW S.; WETHERALL, DAVID J., COMPUTER NETWORKS, 5th Edition, © 2011.
Electronically reproduced by permission of Pearson Education, Inc., Upper Saddle River, New Jersey