

Computer Networks

Flow Control (§6.5.8)



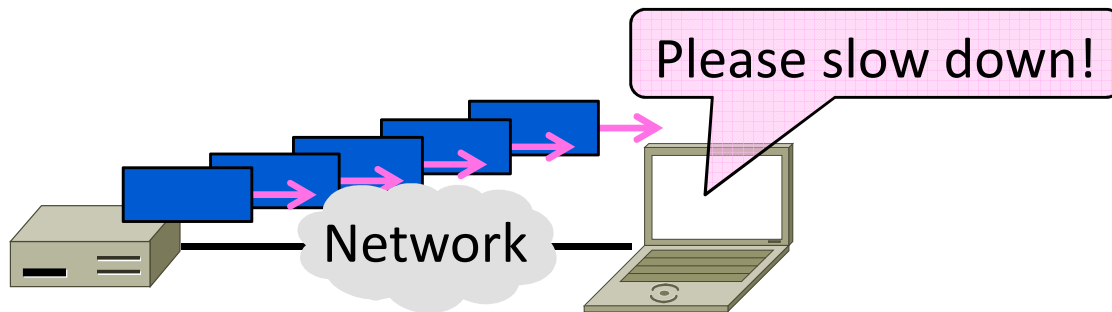
David Wetherall (djw@uw.edu)

Professor of Computer Science & Engineering

UNIVERSITY *of* WASHINGTON

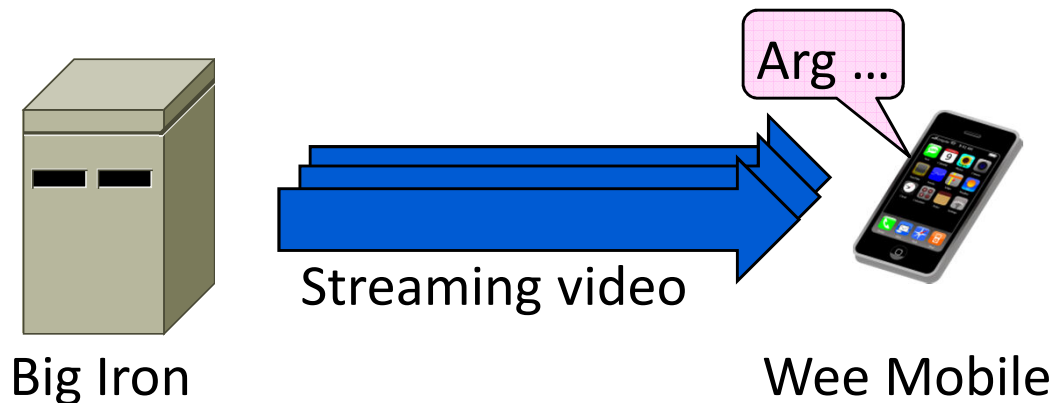
Topic

- Adding flow control to the sliding window algorithm
 - To slow the over-enthusiastic sender



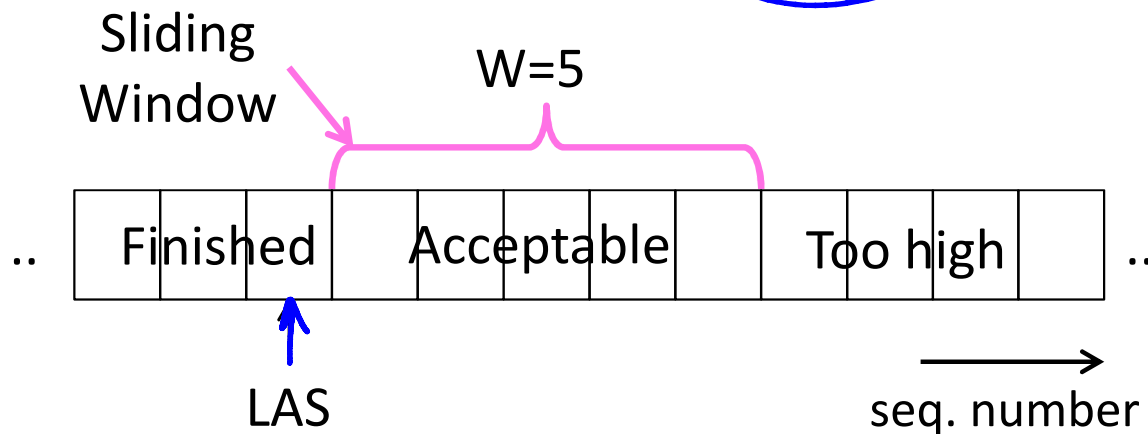
Problem

- Sliding window uses pipelining to keep the network busy
 - What if the receiver is overloaded?



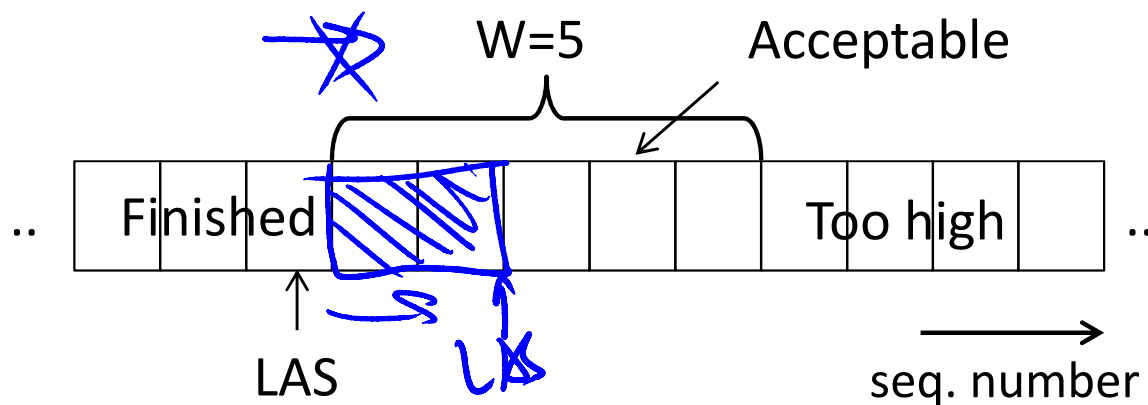
Sliding Window – Receiver

- Consider receiver with W buffers
 - LAS=LAST ACK SENT, app pulls in-order data from buffer with recv() call



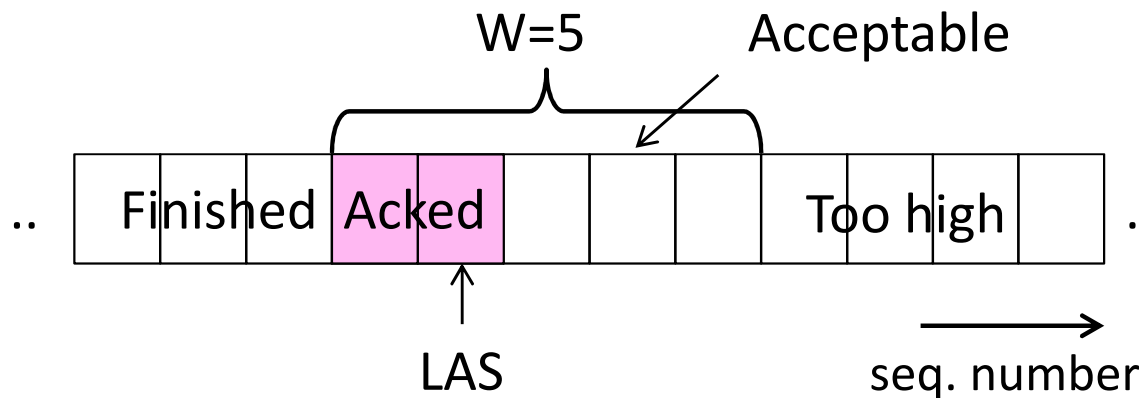
Sliding Window – Receiver (2)

- Suppose the next two segments arrive but app does not call `recv()`



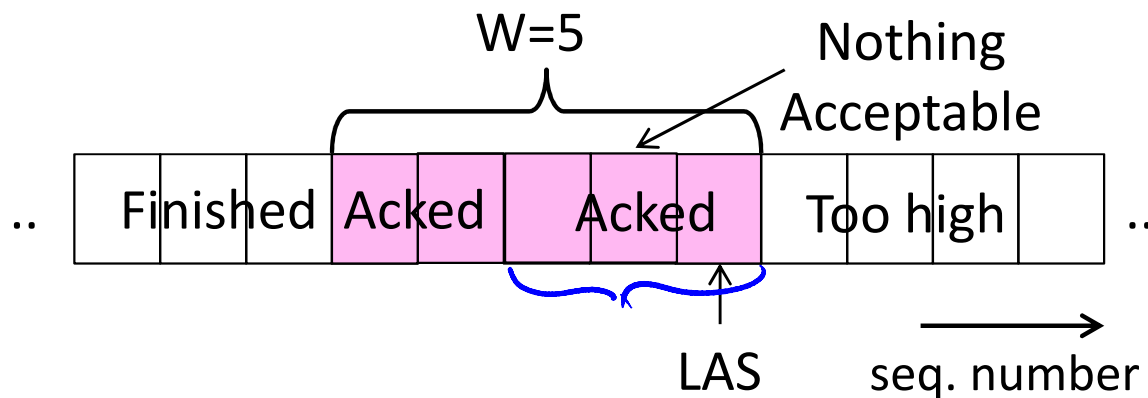
Sliding Window – Receiver (3)

- Suppose the next two segments arrive but app does not call `recv()`
 - LAS rises, but we can't slide window!



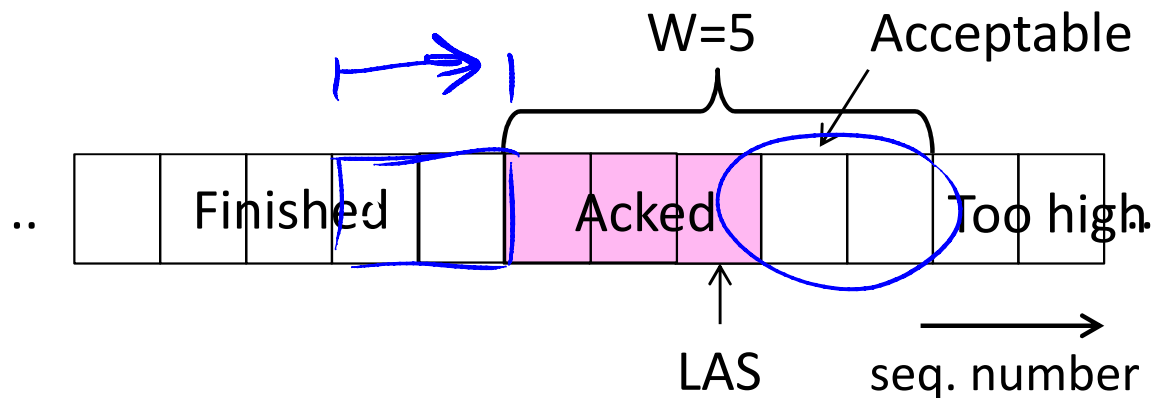
Sliding Window – Receiver (4)

- If further segments arrive (even in order) we can fill the buffer
 - Must drop segments until app recvs!



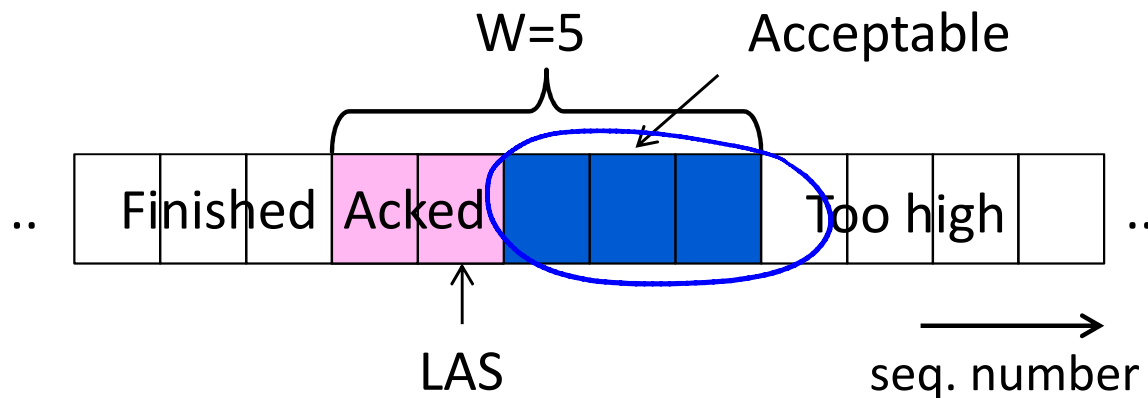
Sliding Window – Receiver (5)

- App recv() takes two segments
 - Window slides (pew)



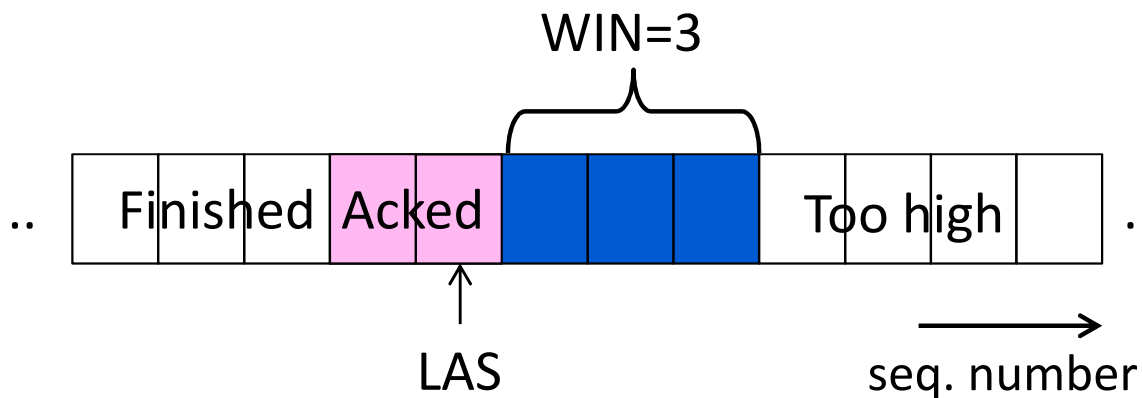
Flow Control

- Avoid loss at receiver by telling sender the available buffer space
 - WIN=#Acceptable, not W (from LAS)



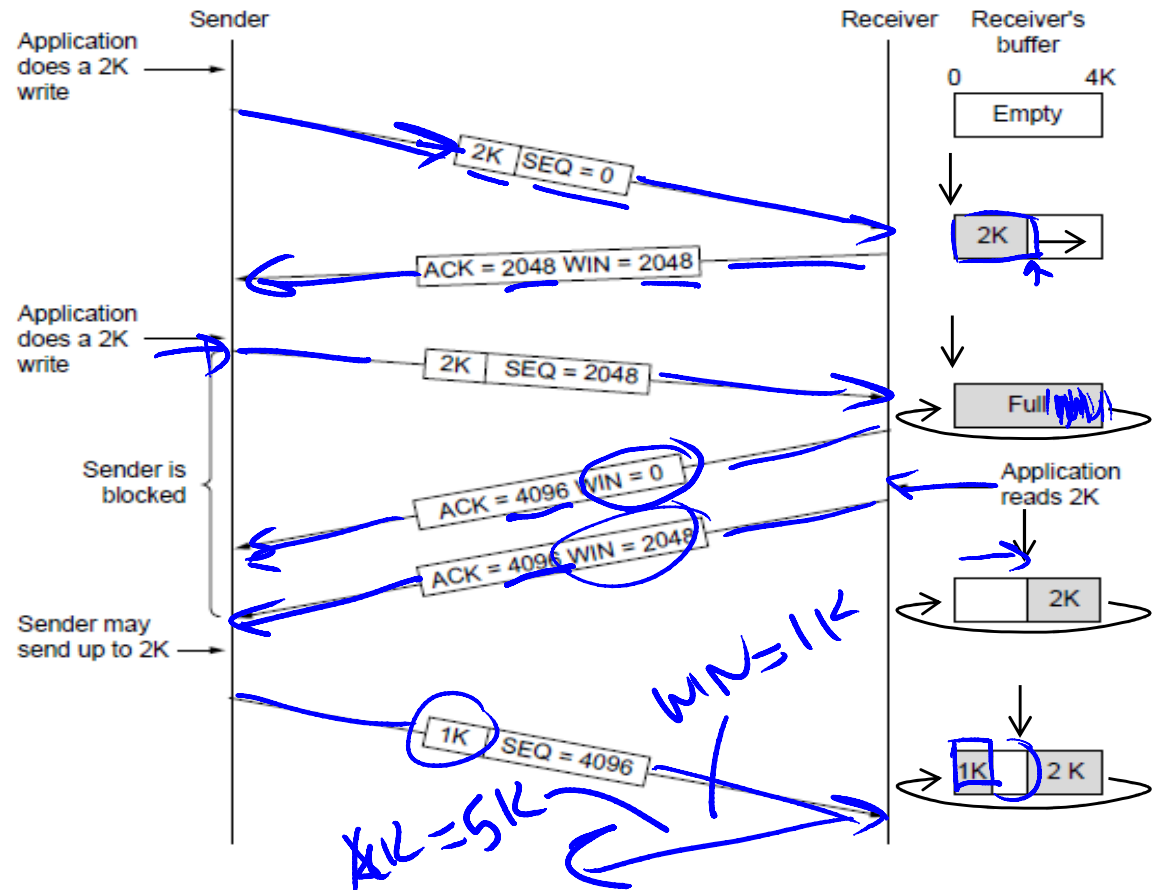
Flow Control (2)

- Sender uses the lower of the sliding window and flow control window (WIN) as the effective window size



Flow Control (3)

- TCP-style example
 - SEQ/ACK sliding window
 - Flow control with WIN
 - $SEQ + \text{length} < ACK + WIN$
 - 4KB buffer at receiver
 - Circular buffer of bytes



END

© 2013 D. Wetherall

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