Stop-and-Wait Algorithm

Timeline showing four different scenarios for the stop-and-wait algorithm.
(a) The ACK is received before the timer expires; (b) the original frame is lost; (c) the ACK is lost; (d) the timeout fires too soon. In (c) and (d), duplicate frames are delivered.

Stop-and-Wait Algorithm

Timeline for stop-and-wait with 1-bit sequence number
Sliding Window Algorithm

Timeline for Sliding Window Algorithm

Sender maintains 3 variables and 1 invariant:
- SWS (Send window size): Upper bound on the number of outstanding (un-ACKed) frames that the sender can transmit
- LAR: last ACK received
- LFS: last frame sent
- Invariant: LFS – LAR ≤ SWS

Sender actions:
- Associate a timer with each transmitted frame; retransmit the frame upon timeout.
- Buffer un-ACKed frames.
- When an ACK of frame K arrives, send up to (K - LAR) frames and set LAR to K
Sliding Window Algorithm: Receiver

Receiver maintains 3 variables and 1 invariant:
- RWS (Receive Window Size): Upper bound on the number of out-of-order frames that the receiver is willing to accept
- LAF: largest acceptable frame
- LFR: last in-order frame received
- Invariant: \( LAF - LFR = RWS \)

Receiver actions:
- If \( LFR < \text{SeqNum} \leq LAF \), accept the frame
  - If the frame is the next frame expected
    - ACK the receipt of \( \text{SeqNumToAck} \), which is the seq no. of the highest numbered frame received in order. *(Cumulative ACK)*
    - Set \( LFR = \text{SeqNumToAck} \) and \( LAF = LFR + RWS \)
  - Else, no ACK needs to be sent. However, resending last ACK allows early detection of frame loss.

Else discard the frame and ACK LFR

The Receiver Algorithm: An Example

- Suppose \( LFR = 5 \) and \( RWS = 4 \), so \( LAF = 9 \)

- If frames 7 and 8 arrive, they will be buffered because they are within the receiver window
  - No ACK will be sent since frame 6 is yet to arrive
  - Frames 7 and 8 are out of order

- When frame 6 arrives (it is late because it was lost first time and had to be retransmitted)
  - ACK 8 is sent
  - LFR is set to 8 and LAF is set to 12
Go-Back-N

Sender:
• Associate a timer with each transmitted frame; upon timeout for frame n, retransmit frame n and all outstanding frames with higher seq no.
• Buffer un-ACKed frames.
• When an ACK of frame K arrives, send up to (K - LAR) frames and set LAR to K

Receiver:
• If SeqNum = LFR + 1, accept the frame
  — ACK the receipt of (LFR+1)
  — LFR = LFR + 1
• Else discard the frame and ACK LFR
• No need to maintain LAF because LAF=LFR+1