Some Flow Control Algorithms

1. Flow control for the ideal network

2. Stop and Wait for noiseless channels

3. Stop and Wait for noisy channels

4. Sliding window protocols

5. Sliding window with error control
   - Go Back N
   - Selective Repeat

1. Flow control in the ideal network

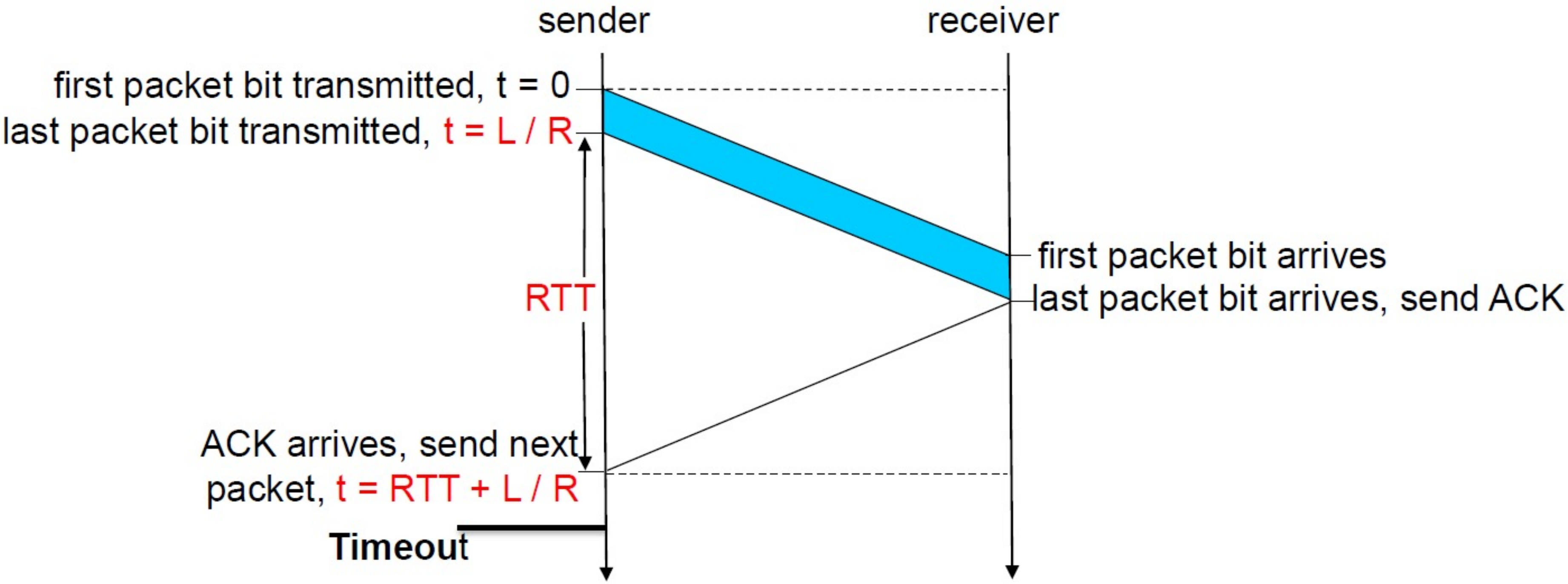
Assumptions:
(1) Error free transmission link,
(2) Infinite buffer at the receiver

No acknowledgement necessary

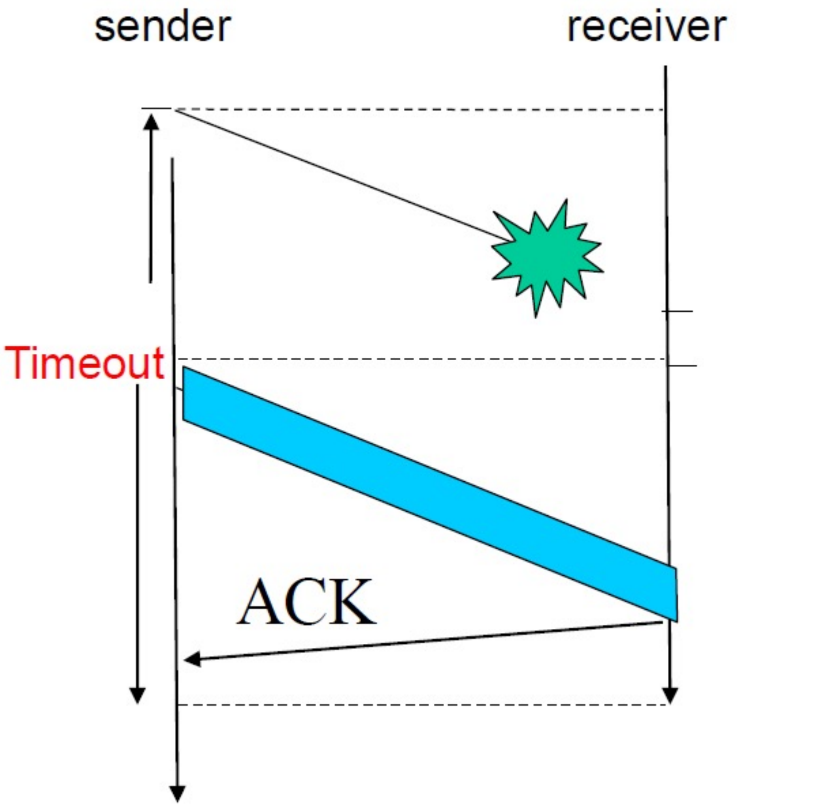
Since the data link is error-free and the receiver can buffer as many packet as it likes, no packet will ever be lost

2. Stop-and-Wait Noiseless Channel

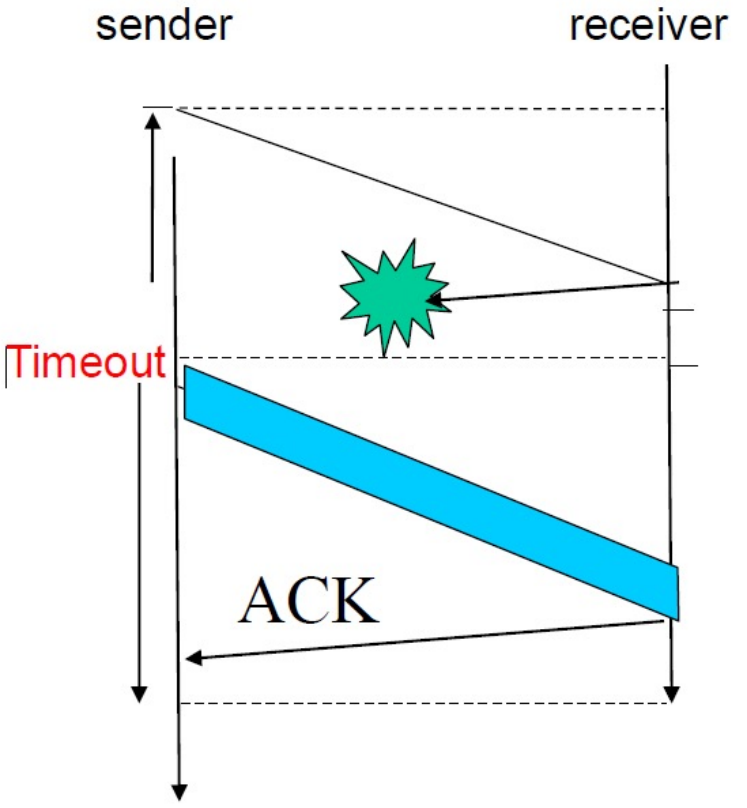
Packet Length = L; Bandwidth =R; RTT = 2*Prop Delay


sender
receiver
first packet bit transmitted, t = 0
last packet bit transmitted, t = L / R
RTT
first packet bit arrives
last packet bit arrives, send ACK
ACK arrives, send next packet, t = RTT + L / R
Timeout

3. Stop-and-Wait Noisy Channel



Packet retransmitted



Packet retransmitted

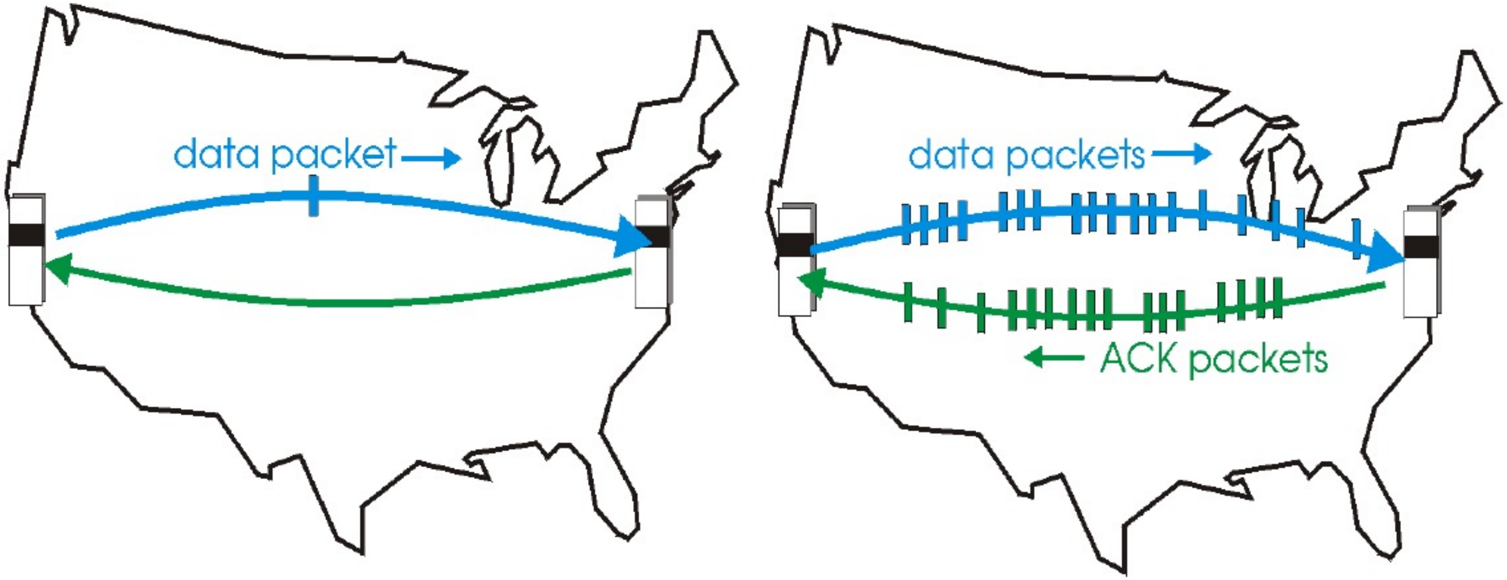
Is Stop and Wait the best we can do?

Stop and Wait is an effective form of flow control, but…

It's not very efficient.

1. Only one data frame can be in transit on the link at a time
2. When waiting for an acknowledgement, the sender cannot transmit any frames

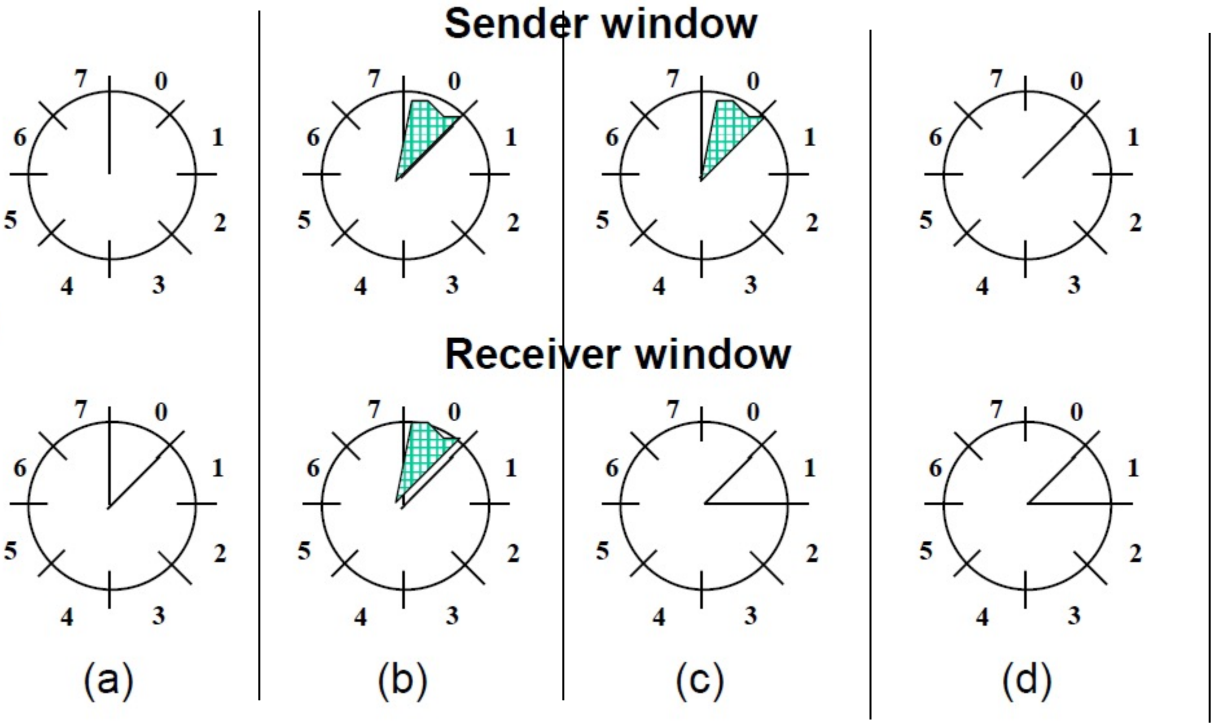
Better solution? Pipelined Protocol : Sliding Window

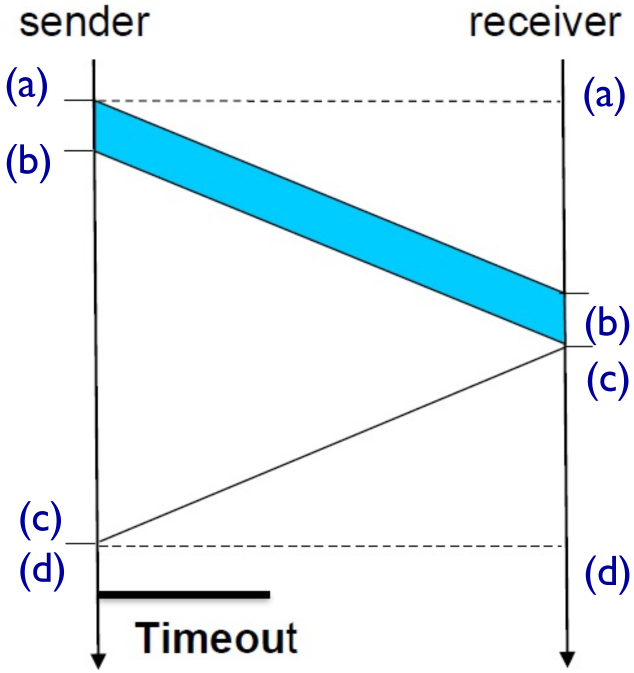


(a) a stop-and-wait protocol in operation

(b) a pipelined protocol in operation

Sliding Window example





(a) Initial state, no frames transmitted, receiver expects frame 0
(b) Sender transmits frame 0, receiver buffers frame 0
(c) Receiver ACKS frame 0
(d) Sender receives ACK, removes frame 0

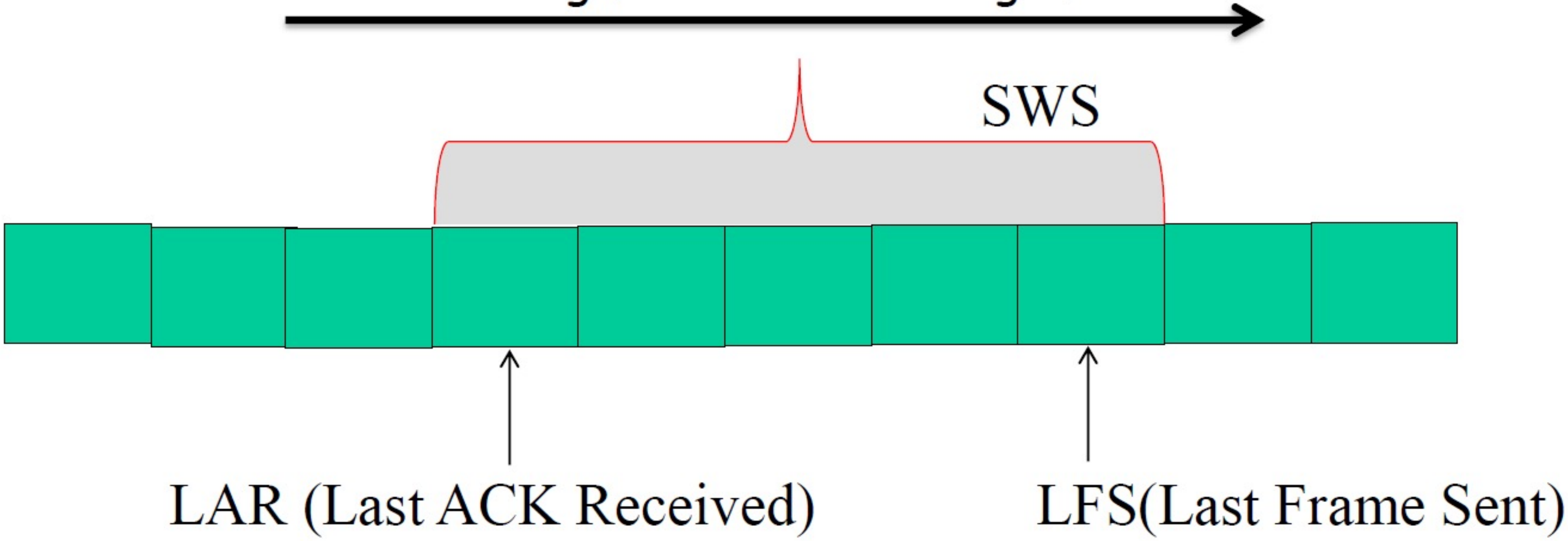
This protocol behaves identically to stop and wait for a noisy channel

Sliding Window with Maximum Sender Window Size *SWS*

Sender Window size: The maximum number of frames the sender may transmit without receiving any acknowledgements

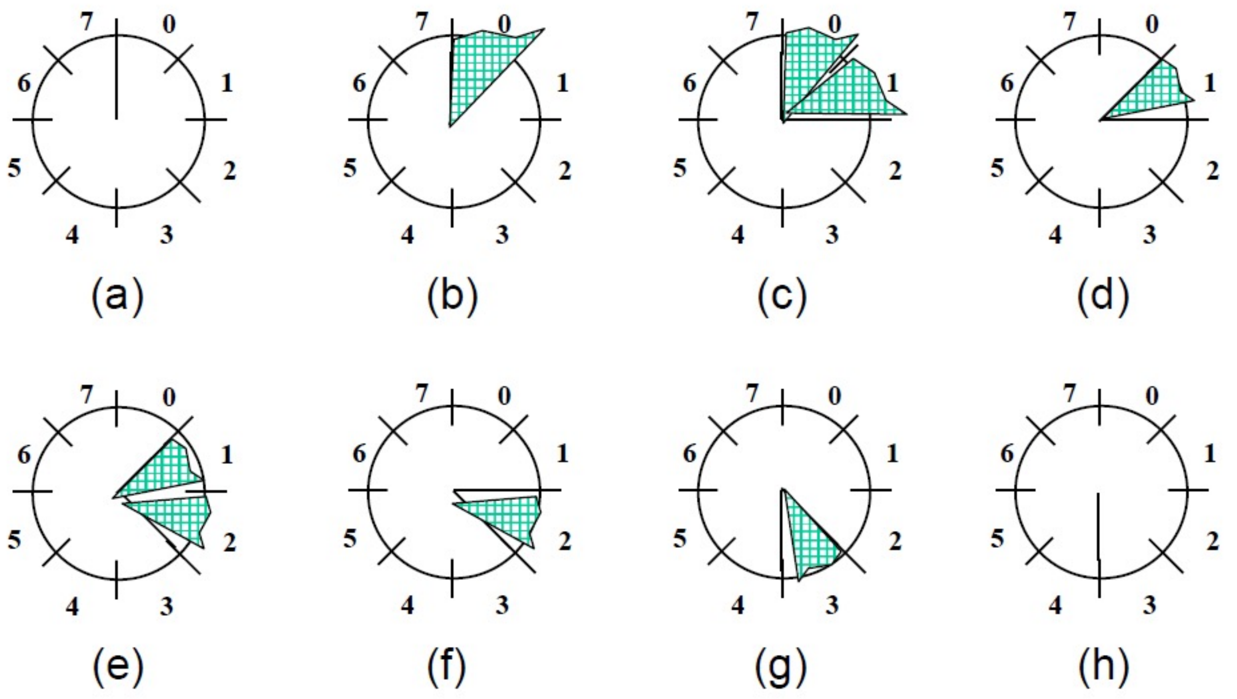
With a maximum window size of *SWS*, the sender can transmit up to *SWS* frames before "being blocked"

This allows the sender to transmit several frames before waiting for an acknowledgement



LFS - LAR <= SWS

Sender-Side Window with W_S=2



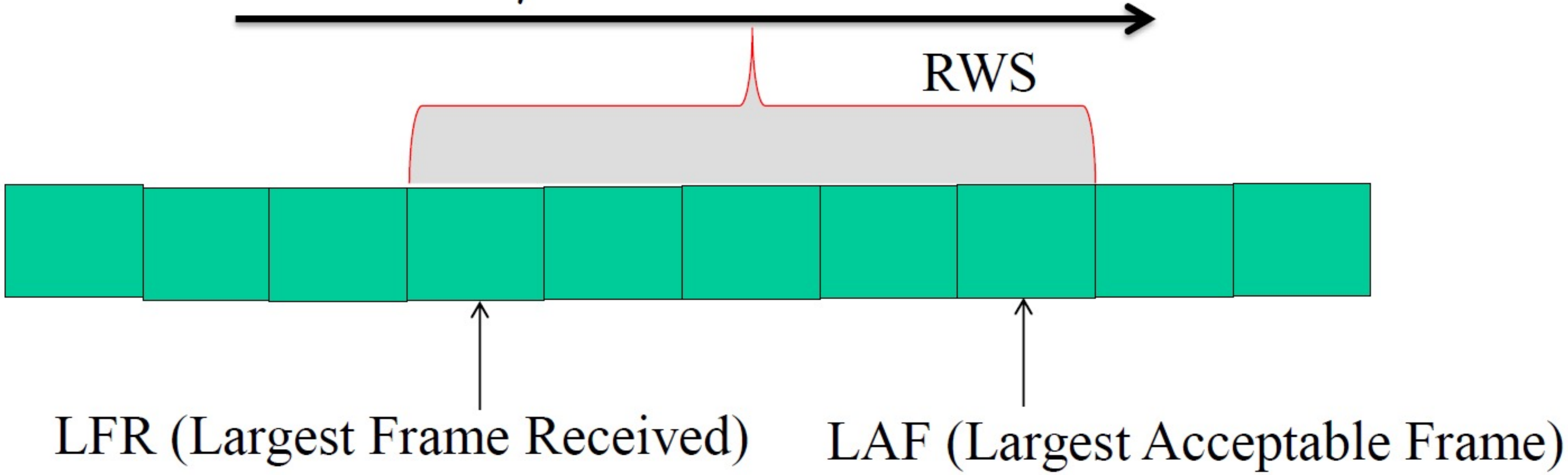
(a) Initial window state
(b) Send frame 0
(c) Send frame 1
(d) ACK for frame 0 arrives
(e) Send frame 2
(f) ACK for frame 1 arrives
(g) ACK for frame 2 arrives, send frame 3
(h) ACK for frame 3 arrives

Sliding Window with Maximum Receiver Window Size

Receiver Window size: The maximum number of frames the receiver may receive before returning an acknowledgement to the sender

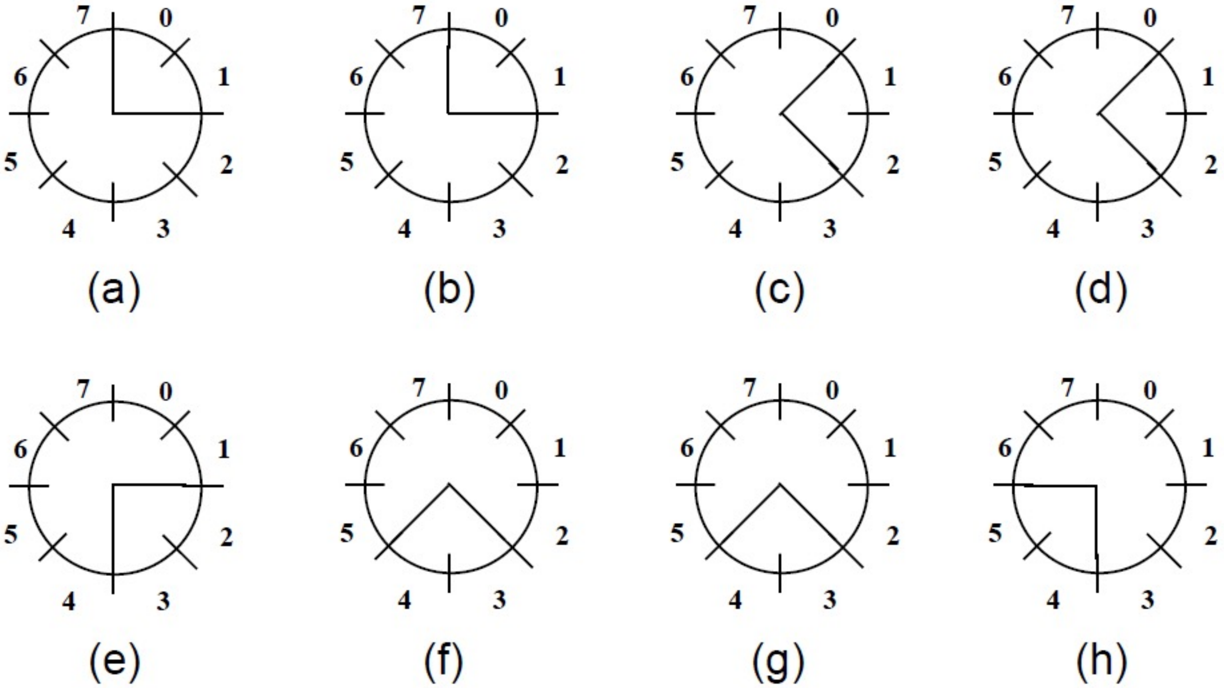
With a maximum window size of *RWS*, the receiver rejects packets if SeqNum <= LFR or SeqNum > LAF

Why? Outside the window



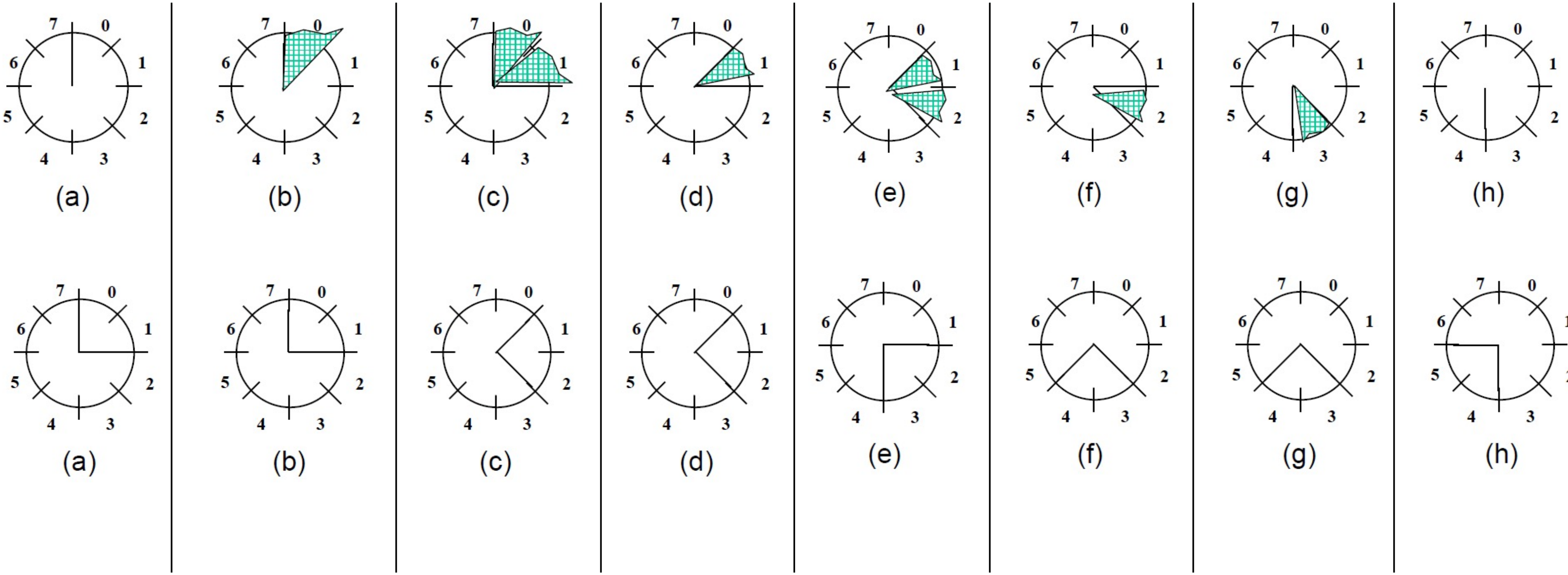
$$LAF - LFR <= RWS$$

Receiver-Side Window with W_R=2



(a) Initial window state
(b) Nothing happens
(c) Frame 0 arrives, ACK frame 0
(d) Nothing happens
(e) Frame 1 arrives, ACK frame 1
(f) Frame 2 arrives, ACK frame 2
(g) Nothing happens
(h) Frame 3 arrives, ACK frame 3

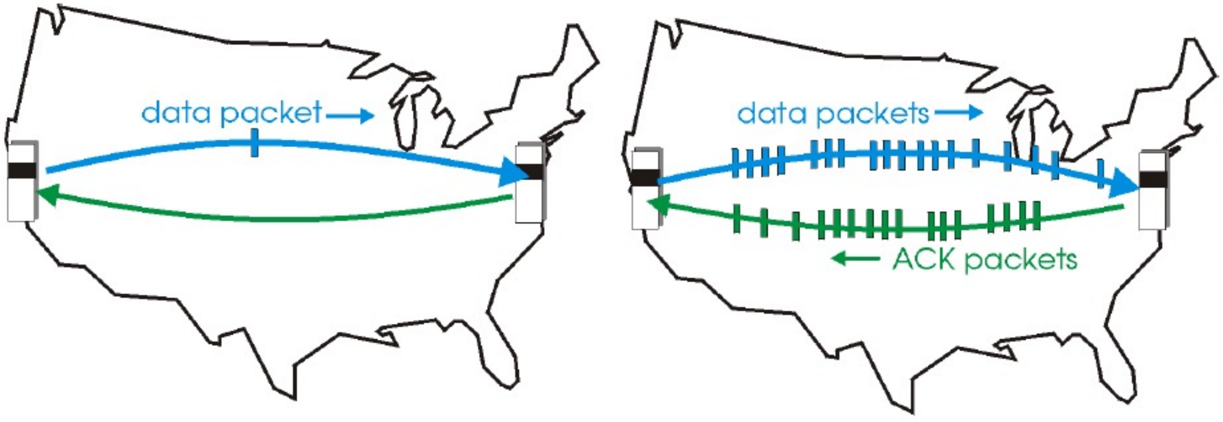
Sender-Side Window with W_S=2



Receiver-Side Window with W_R=2

What about Errors?

What if a data or acknowledgement frame is lost when using a sliding window protocol?



(a) a stop-and-wait protocol in operation

(b) a pipelined protocol in operation

Two Solutions:

Go Back N

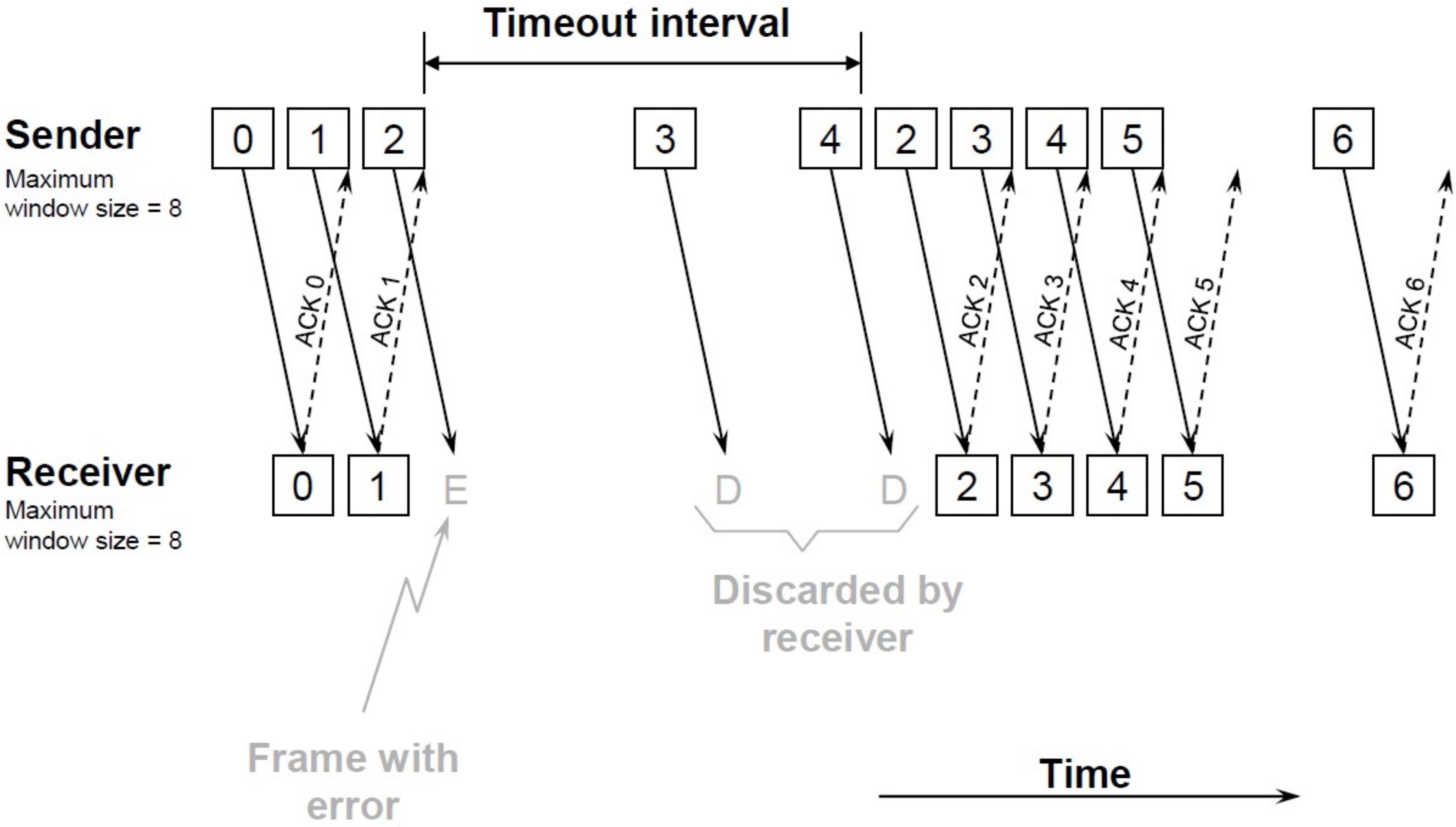
Selective Repeat

Sliding Window with Go Back N

- When the receiver notices a missing or erroneous frame, it simply discards all frames with greater sequence numbers and sends no ACK

- The sender will eventually time out and retransmit all the frames in its sending window

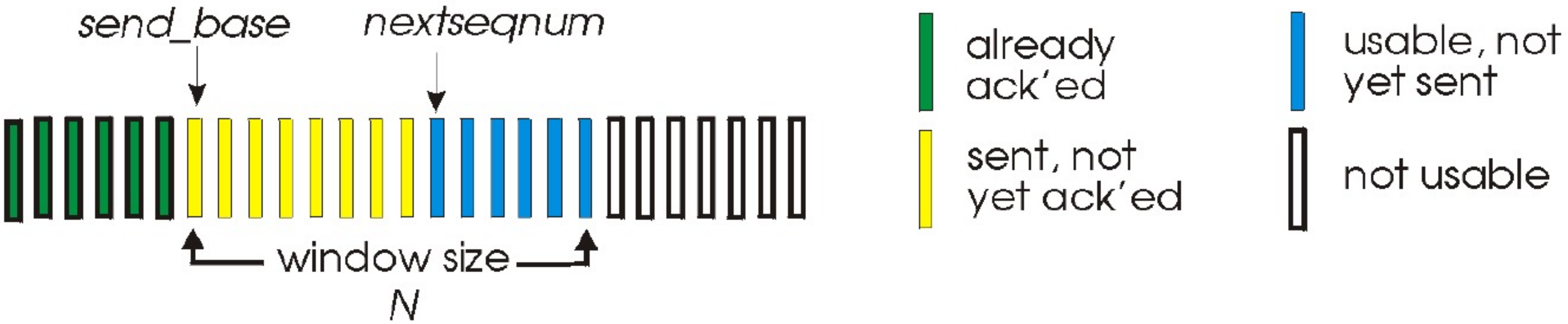
Go Back N


Timeout interval
Sender
Maximum
window size = 8
0
1
2
3
4
2
3
4
5
6
ACK 0
ACK 1
ACK 2
ACK 3
ACK 4
ACK 5
ACK 6
Receiver
Maximum
window size = 8
0
1
E
D
D
2
3
4
5
6
Discarded by
receiver
Frame with
error
Time

Go-Back-N

Sender:

- k-bit seq # in pkt header
- "window" of up to N, consecutive unack'ed pkts allowed



- ACK(n): ACKs all pkts up to seq # n - "cumulative ACK"
- *timeout(n)*: retransmit pkt n and all higher seq # pkts in window
- One timer for all in-flight pkts

Go Back N *(cont'd)*

Go Back N can recover from erroneous or missing frames

But…

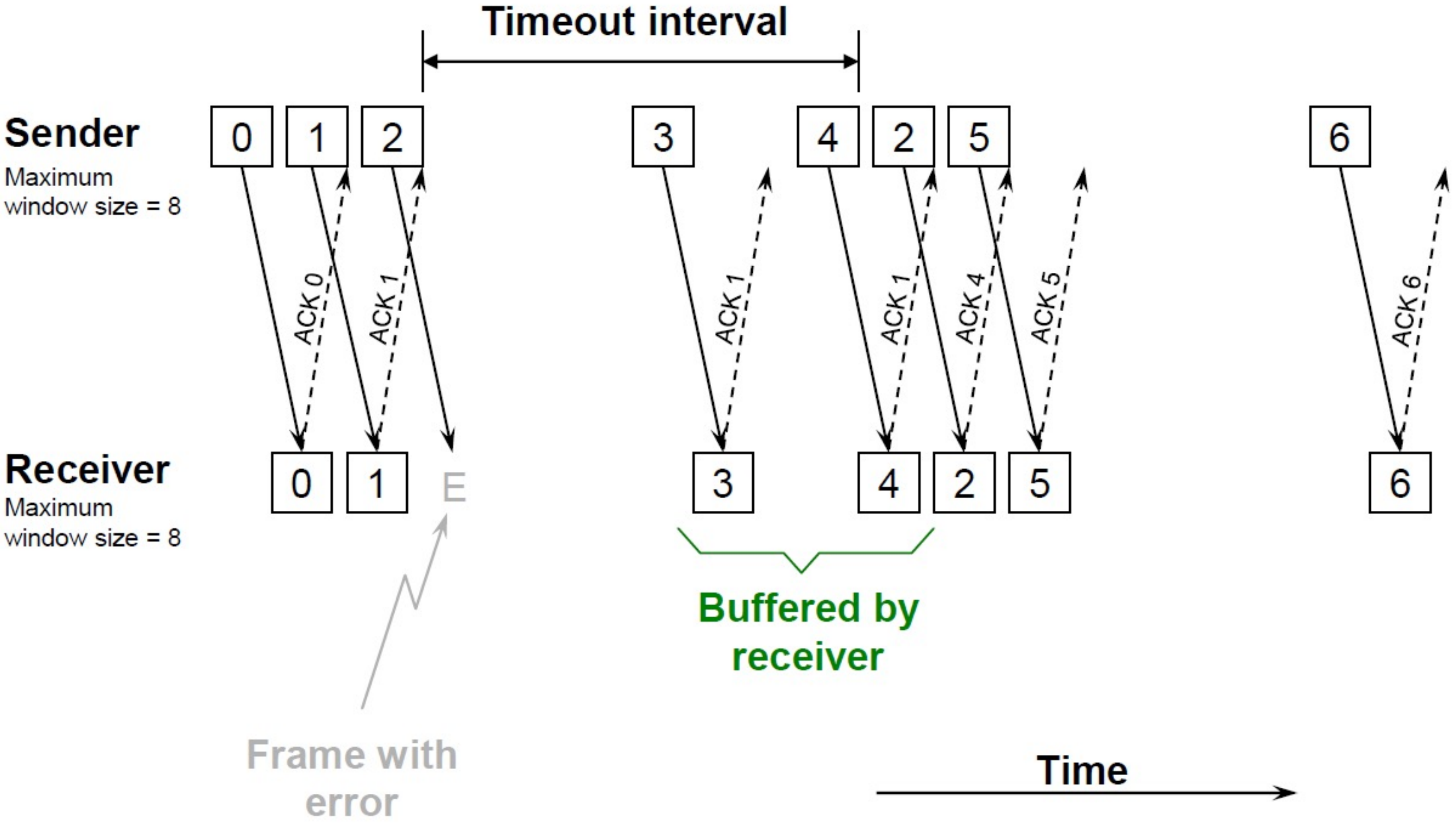
It is wasteful. If there are errors, the sender will spend time retransmitting frames the receiver has already seen

Sliding Window with Selective Repeat

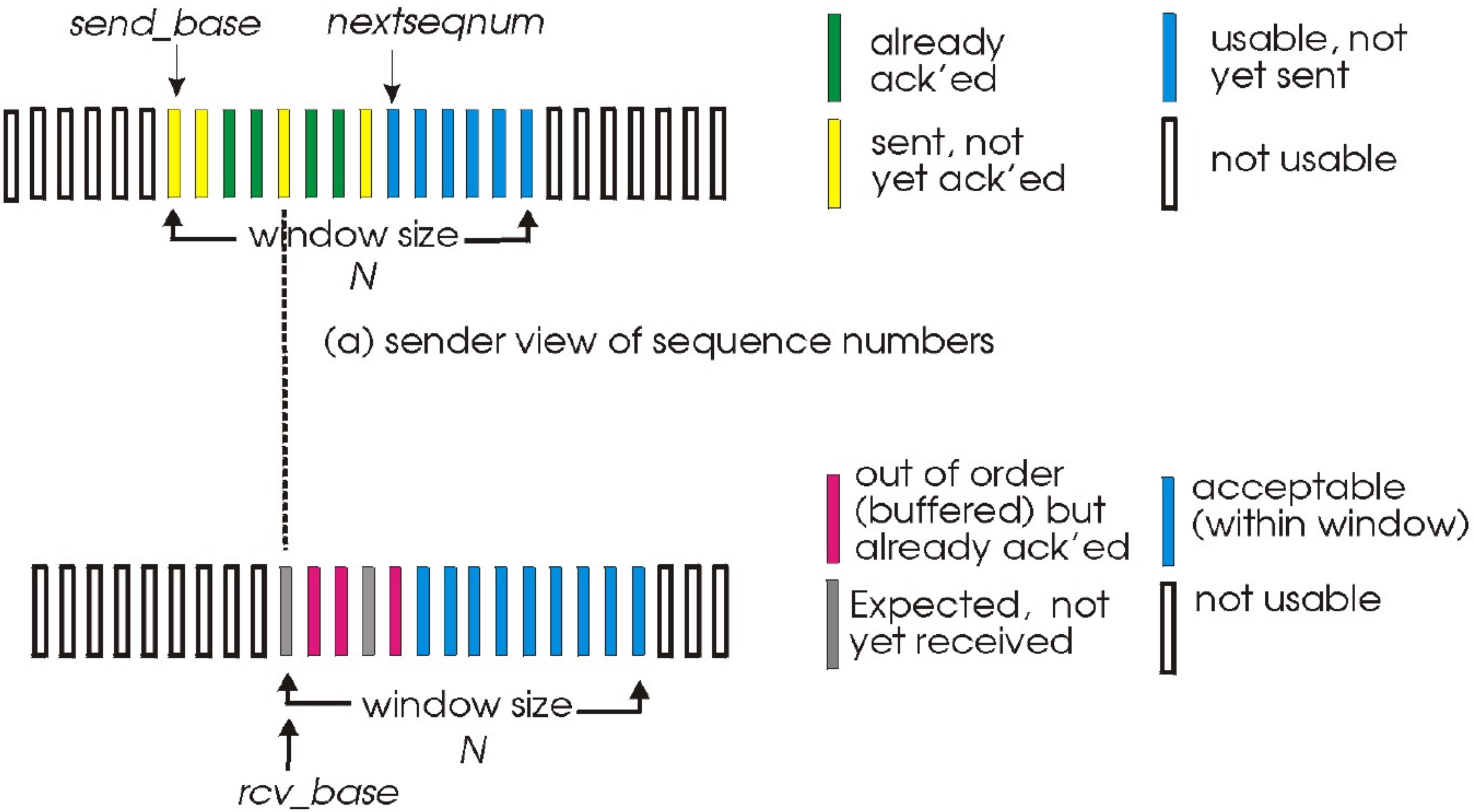
The sender retransmits only the frame with errors

- ❑ The receiver stores all the correct frames that arrive following the bad one.
  - ❖ The receiver requires a frame buffer for each sequence number in its receiver window.
- ❑ When the receiver notices a skipped sequence number, it keeps acknowledging the last good sequence number
- ❑ When the sender times out waiting for an acknowledgement, it just retransmits the one unacknowledged frame, not all its successors.

Selective Repeat



Selective repeat: sender, receiver windows



(a) sender view of sequence numbers

(b) receiver view of sequence numbers

TCP (Transmission Control Protocol)

- TCP provides the end-to-end reliable connection

- The protocol
  1. Connection management
  2. Retransmission
  3. Flow control
  4. Congestion control
  5. Frame format