National University of Computer & Emerging Sciences

CS 3001 - COMPUTER NETWORKS

Lecture 12

Chapter 3

3rd October, 2023

Nauman Moazzam Hayat

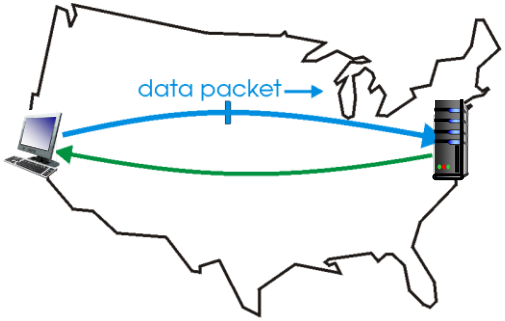
nauman.moazzam@lhr.nu.edu.pk

Office Hours: 02:30 pm till 06:00 pm (Every Tuesday & Thursday)

rdt3.0: pipelined protocols operation

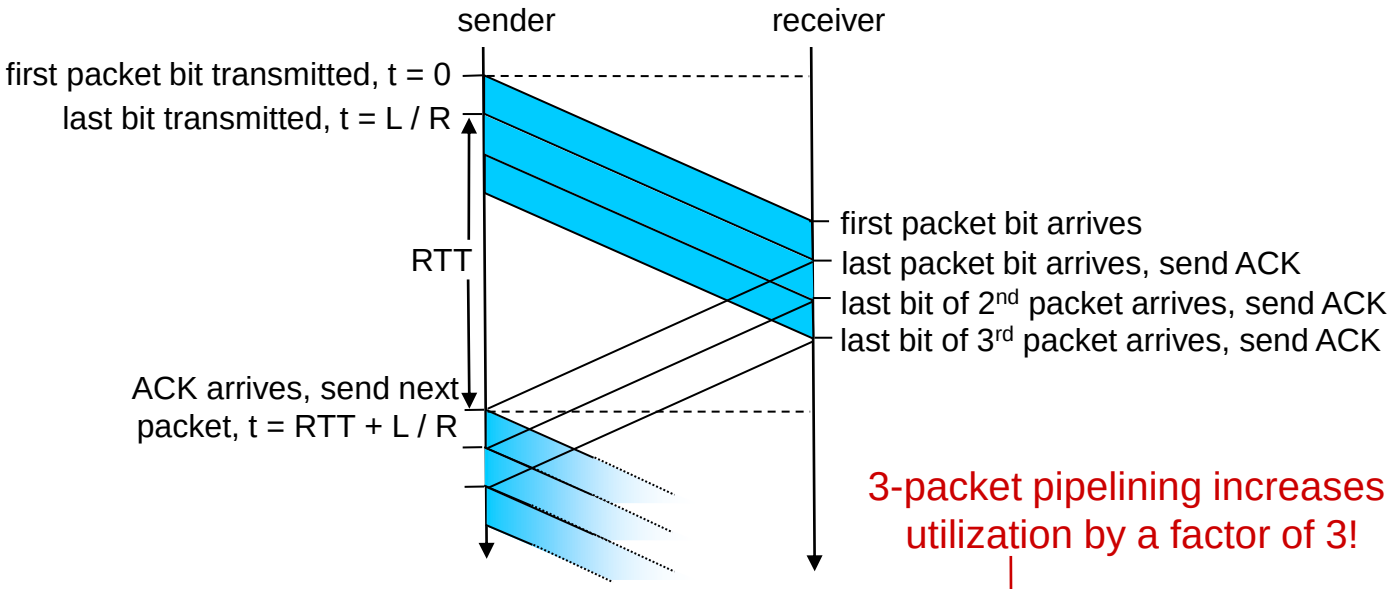
pipelining: sender allows multiple, “in-flight”, yet-to-be-acknowledged packets

- range of sequence numbers must be increased
- buffering at sender and/or receiver



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

Pipelining: increased utilization

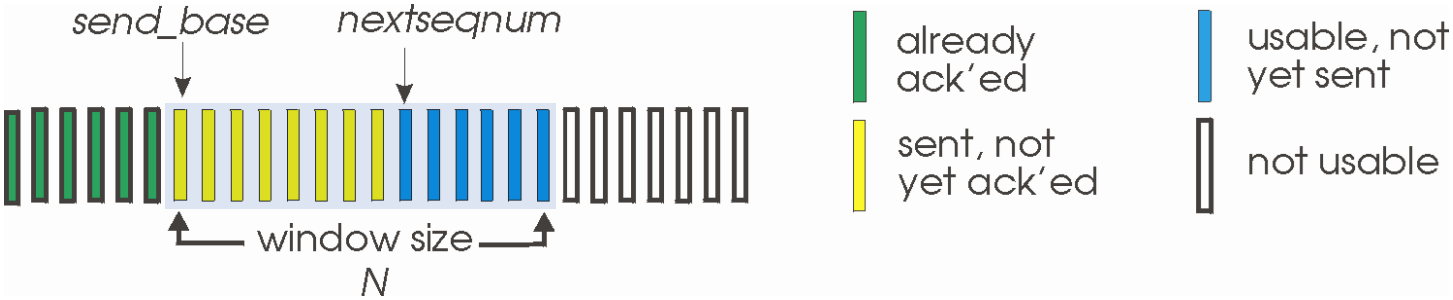


3-packet pipelining increases utilization by a factor of 3!

$$U_{sender} = \frac{3L / R}{RTT + L / R} = \frac{.0024}{30.008} = 0.00081$$

Go-Back-N: sender

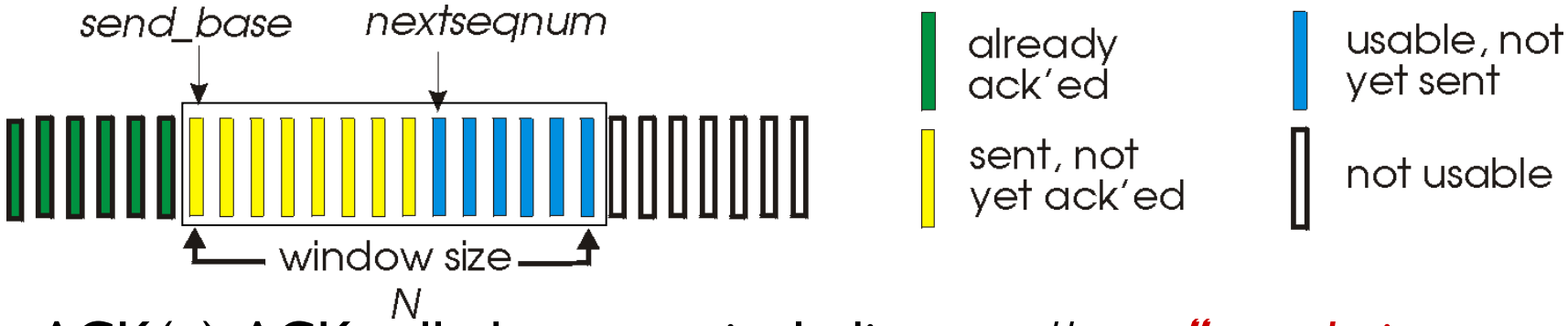
- sender: "window" of up to N, consecutive transmitted but unACKed pkts
  - k-bit seq # in pkt header



- *cumulative ACK:* ACK(*n*): ACKs all packets up to, including seq # *n*
  - on receiving ACK(*n*): move window forward to begin at *n+1*
- timer for oldest in-flight packet
- *timeout(n):* retransmit packet n and all higher seq # packets in window

Go-Back-N: sender (Sliding Window)

- k-bit seq # in pkt header, range of sequence numbers is $[0, 2^k - 1]$
- sliding "window" of up to N, consecutive unack'ed pkts allowed
- N should be always greater than 1 in order to implement pipelining. For N = 1, it reduces to Stop and Wait protocol.(Min sequence numbers required in GBN=N+1)

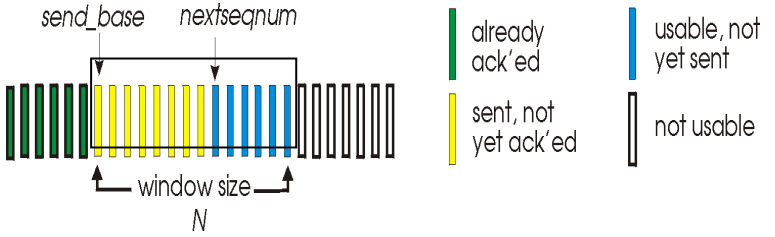


- ACK(n): ACKs all pkts up to, including seq # n - *"cumulative ACK"* (i.e. One ACK is used for many packets. The main advantage is that traffic is less. A disadvantage is less reliability i.e. if one ack is lost, that would mean that all the packets sent are lost.)
  - may receive duplicate ACKs (see receiver)
- timer for oldest in-flight pkt
- *timeout(n):* retransmit packet n and all higher seq # pkts in window

Go-Back-N: sender

- Invocation from above
  - When rdt_sent() is called, checks if window is full
  - If not full, a packet is created and sent
- Receipt of an ACK
  - Cumulative acknowledgement: ack with seq number n means all previous packets has been received at receiver
- A timeout event
  - All previous packets that have been sent, their acknowledgements have not been received will be resent

4 Ranges of Sequence Numbers

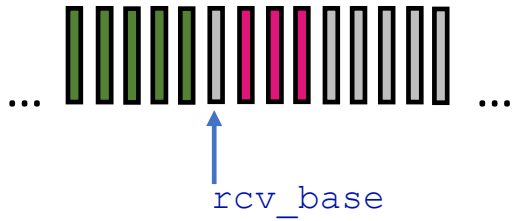


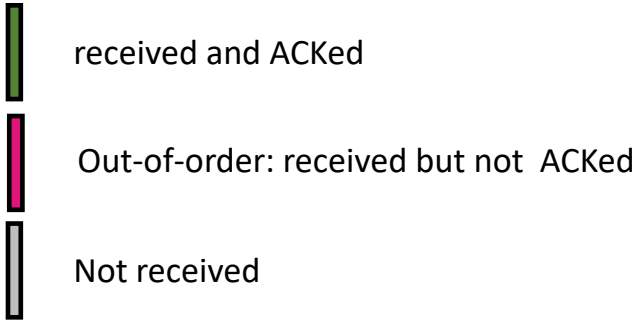
1. [0, base -1] → are packets with sequence numbers sent & ACKed.
2. [base, nextseqnum - 1] → are sequence numbers sent but not yet ACKed (inflight.)
3. [nextseqnum, base + N - 1] → are sequence numbers that can be sent immediately if more data arrives from the App layer.
4. [>=, base + N] → are sequence numbers that can't be used until an unACKed packet is ACKed.

Go-Back-N: receiver

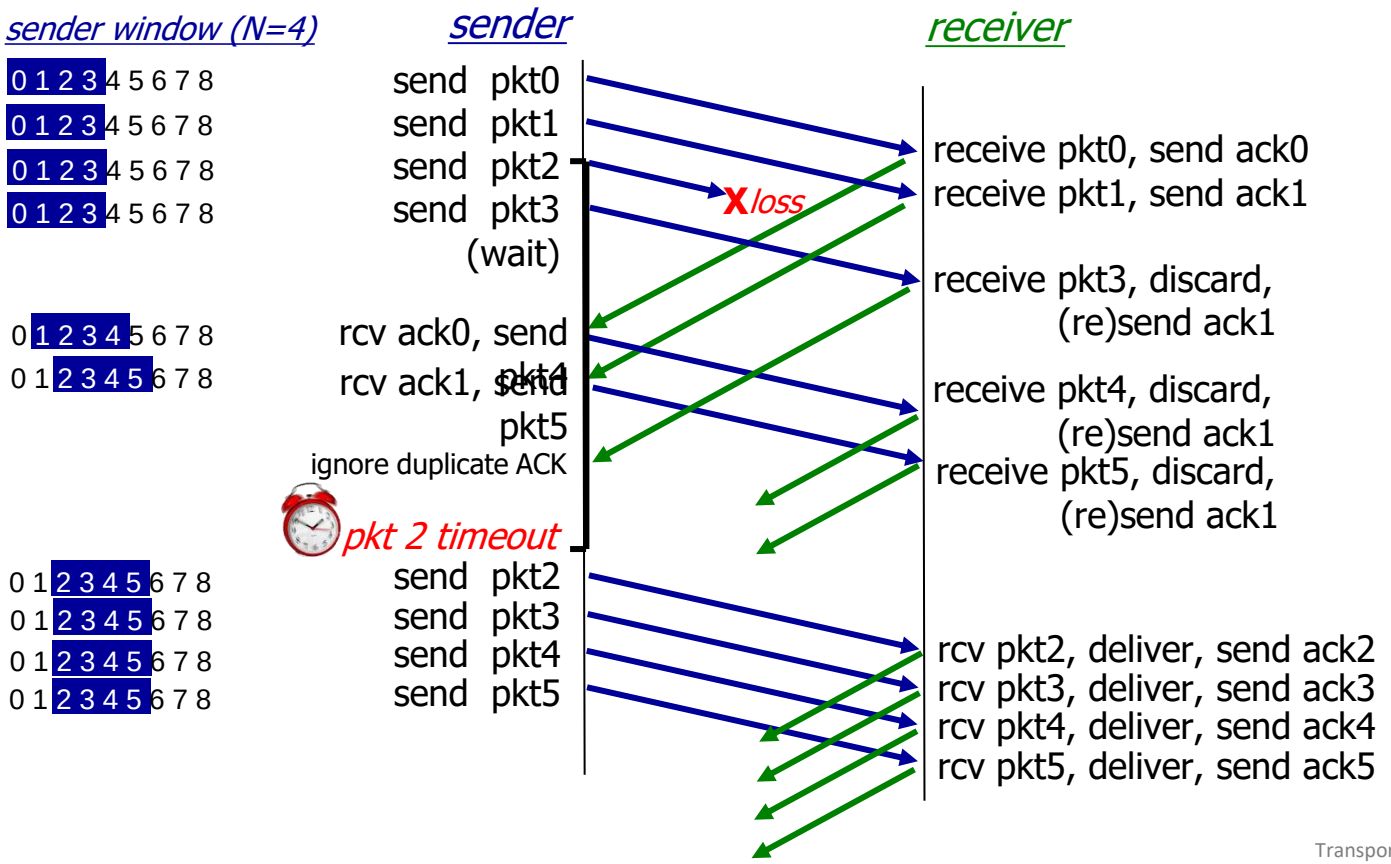
- ACK-only: always send ACK for correctly-received packet so far, with highest *in-order* seq #
  - may generate duplicate ACKs
  - need only remember `rcv_base`
- on receipt of out-of-order packet:
  - can discard (don't buffer) or buffer: an implementation decision
  - re-ACK pkt with highest in-order seq #

Receiver view of sequence number space:





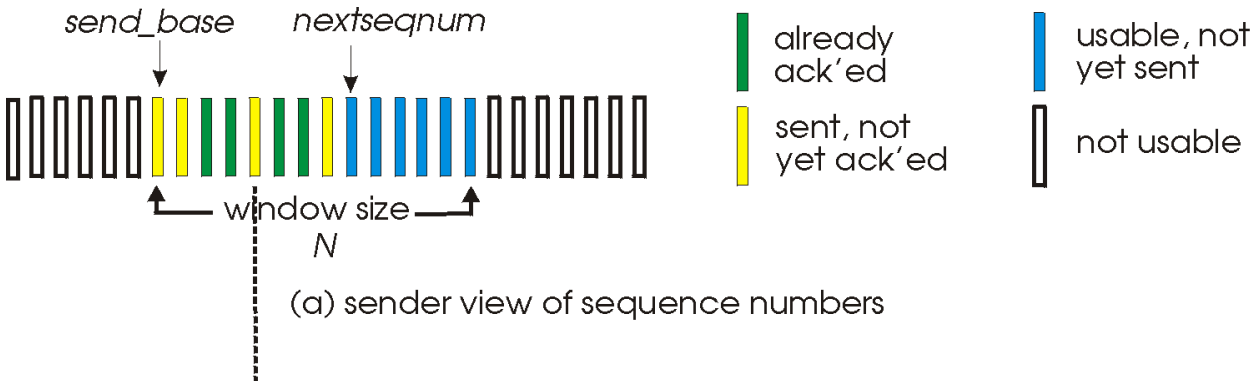
Go-Back-N in action



Selective repeat: the approach

- *pipelining*: *multiple* packets in flight
- *receiver individually ACKs* all correctly received packets
  - buffers packets, as needed, for in-order delivery to upper layer
- sender:
  - maintains (conceptually) a timer for each unACKed pkt
    - timeout: retransmits single unACKed packet associated with timeout
  - maintains (conceptually) "window" over *N* consecutive seq #s
    - limits pipelined, "in flight" packets to be within this window

Selective repeat: sender, receiver windows



Sender window size = Receiver window size =N

Selective repeat: sender and receiver

sender

data from above:

- if next available seq # in window, send packet

timeout(*n*):

- resend packet *n*, restart timer

ACK(*n*) in [sendbase,sendbase+N-1]:

- mark packet *n* as received
- if n smallest unACKed packet, advance window base to next unACKed seq #

receiver

packet *n* in [rcvbase, rcvbase+N-1]

- send ACK(*n*)
- out-of-order: buffer
- in-order: deliver (also deliver buffered, in-order packets), advance window to next not-yet-received packet

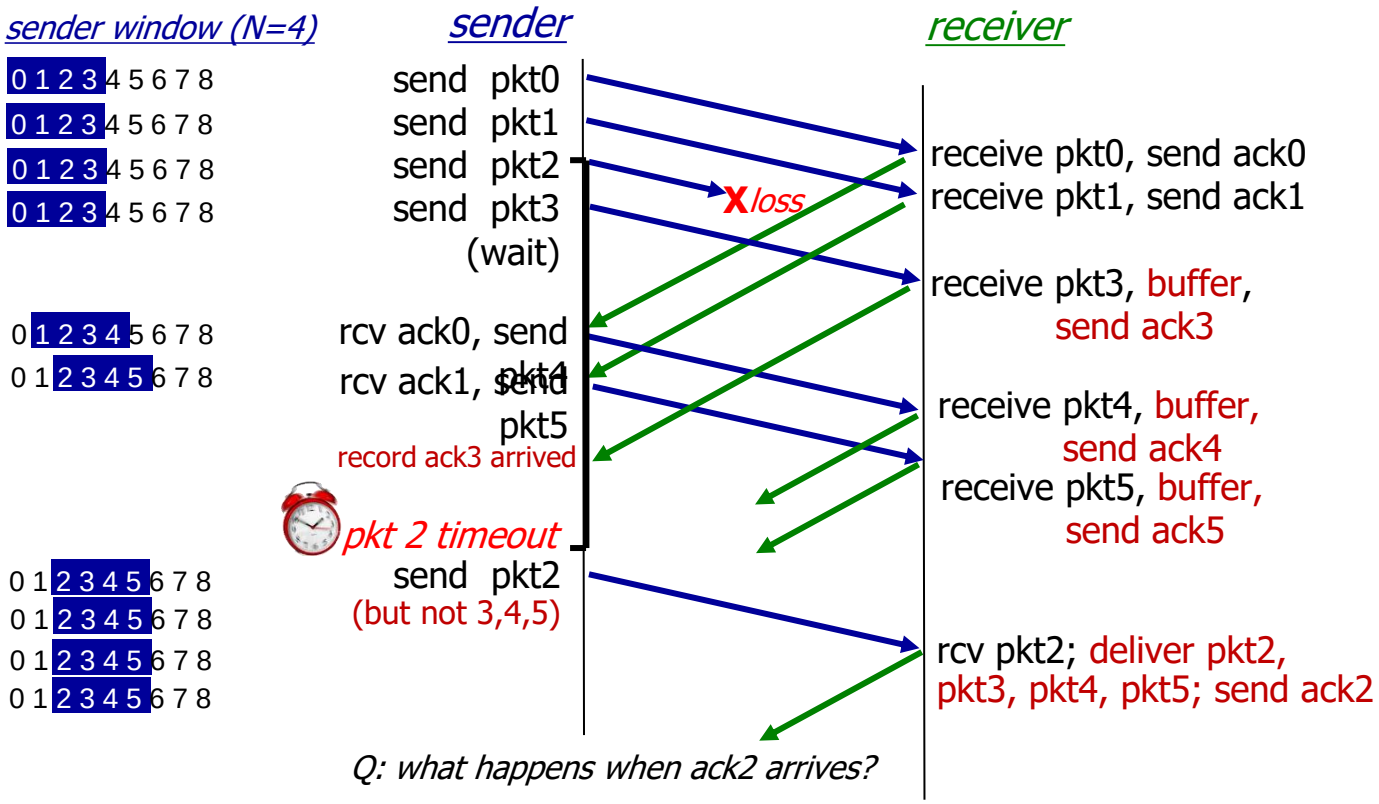
packet *n* in [rcvbase-N,rcvbase-1]

- ACK(*n*)

otherwise:

- ignore

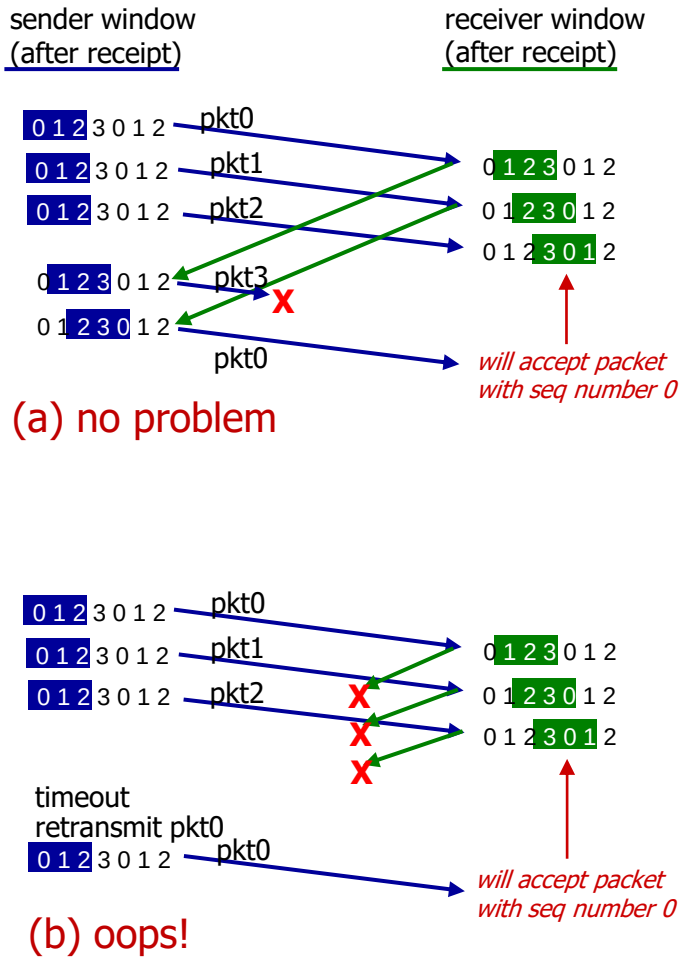
Selective Repeat in action



Selective repeat: a dilemma!

example:

- seq #s: 0, 1, 2, 3 (base 4 counting)
- window size=3

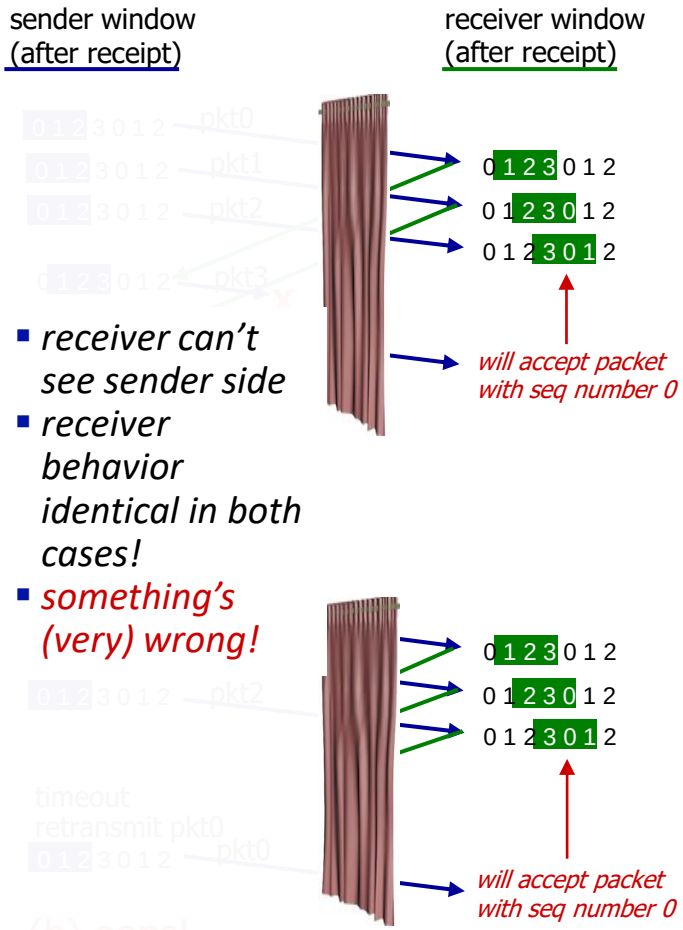


Selective repeat: a dilemma!

example:

- seq #s: 0, 1, 2, 3 (base 4 counting)
- window size=3

Q: what relationship is needed between sequence # size and window size to avoid problem in scenario (b)?



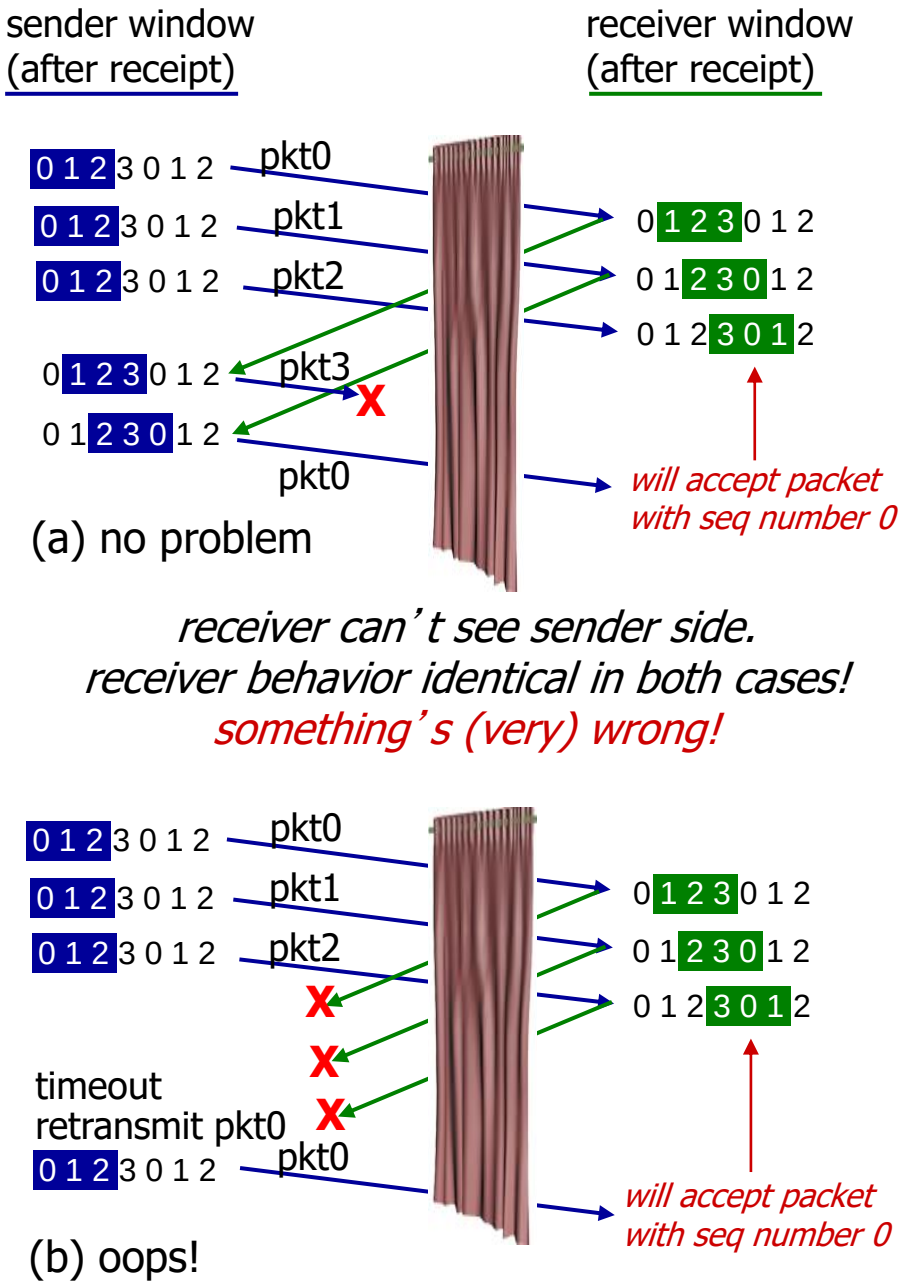
Selective repeat: dilemma

example:

- seq #'s: 0, 1, 2, 3
- window size=3
- receiver sees no difference in two scenarios!
- duplicate data accepted as new in (b) (new packet or retransmission?)

Q: what relationship between seq # size and window size to avoid problem in (b)?

Window size should be less than or equal to half the sequence number space in SR protocol. This is to avoid packets being recognized incorrectly. If the windows size is greater than half the sequence number space, then if an ACK is lost, the sender may send new packets that the receiver believes are retransmissions.



Major Differences between Stop & Wait, Go-Back-N (GBN) and Selective Repeat (SR) [1/2]

- *Stop and Wait*

The sender sends the packet and waits for the ACK (acknowledgement) of the packet. Once the ACK reaches the sender, it transmits the next packet in row. If the ACK is not received, it re-transmits the previous packet again.

- *Go Back N*

The sender sends N packets which is equal to the window size. Once the entire window is sent, the sender then waits for a cumulative ACK to send more packets. On the receiver end, it receives only in-order packets and discards out-of-order packets. As in case of packet loss, the entire window would be re-transmitted.

- *Selective Repeat*

The sender sends packet of window size N and the receiver acknowledges all packet whether they were received in order or not. In this case, the receiver maintains a buffer to contain out-of-order packets and sorts them. The sender selectively re-transmits the lost packet and moves the window forward.

URL for Interactive Animations:

https://wps.pearsoned.com/ecs_kurose_compnetw_6/216/55463/14198702.cw/index.html

Differences between Stop & Wait, Go-Back-N (GBN) and Selective Repeat (SR) [2/2]

PROPERTIES	STOP AND WAIT	GO BACK N	SELECTIVE REPEAT
Sender window size	1	N	N
Receiver Window size	1	1	N
Minimum Sequence number	2	N+1	2N
Efficiency	1/(1+2*a)	N/(1+2*a)	N/(1+2*a)
Type of Acknowledgement	Individual	Cumulative	Individual
Supported order at Receiving end	–	In-order delivery only	Out-of-order delivery as well
Number of retransmissions in case of packet drop	1	N	1

Where,

- ***a = Ratio of Propagation delay and Transmission delay***
- ***At N=1, Go Back N is effectively reduced to Stop and Wait***
- ***As Go Back N acknowledges the packets cumulatively, it rejects out-of-order packets***
- ***As Selective Repeat supports receiving out-of-order packets (it sorts the window after receiving the packets), it uses Independent Acknowledgement to acknowledge the packets.***