



19CSE301

COMPUTER NETWORKS

3-0-0 3

Amrita Vishwa Vidyapeetham
Amritapuri Campus



TRANSPORT LAYER



Chapter 3: Transport Layer

All material copyright 1996-2016

J.F Kurose and K.W. Ross, All Rights Reserved

Chapter 3 outline

3.1 transport-layer services

3.2 multiplexing and demultiplexing

3.3 connectionless transport: UDP

3.4 principles of reliable data transfer

3.5 connection-oriented transport: TCP

- segment structure
- reliable data transfer
- flow control
- connection management

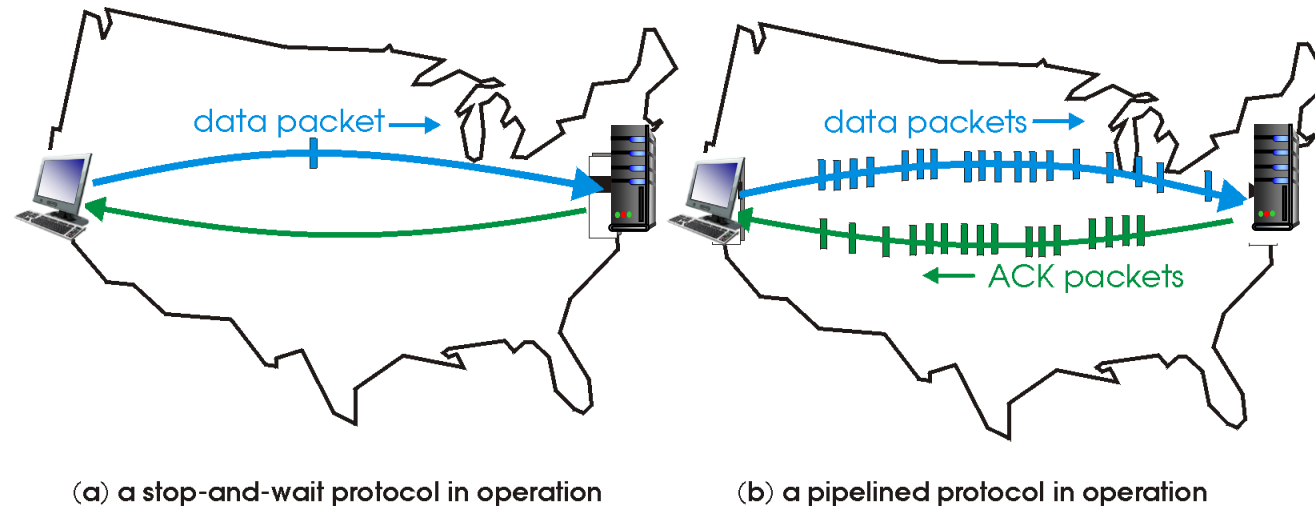
3.6 principles of congestion control

3.7 TCP congestion control

Pipelined protocols

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

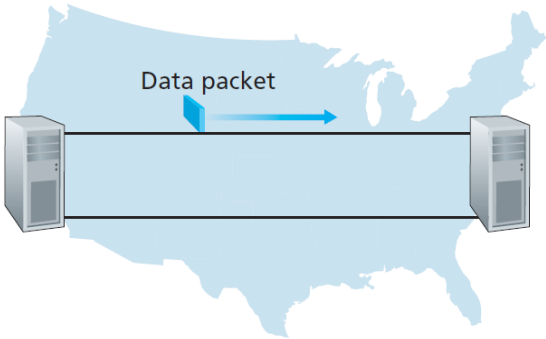
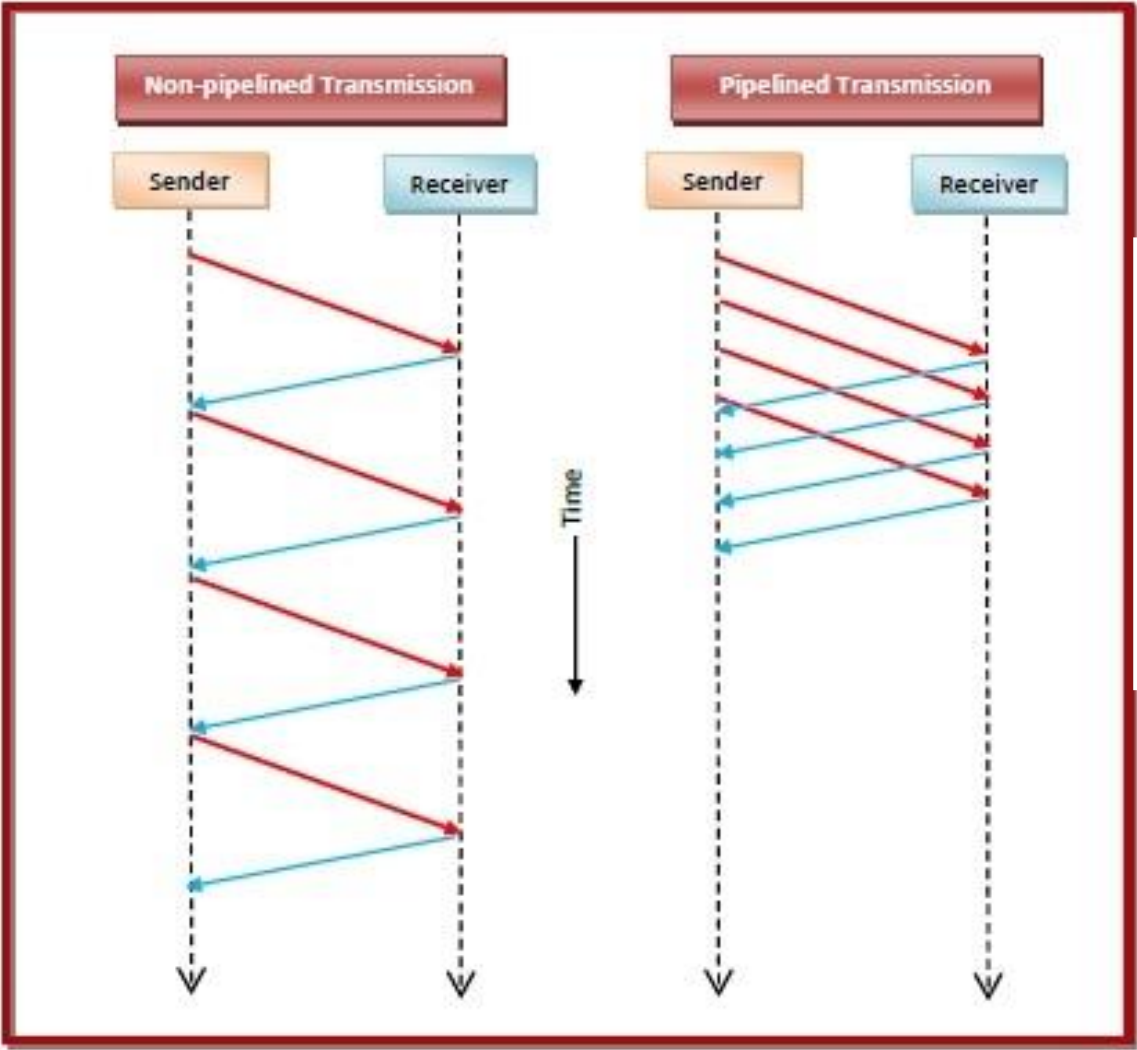
- range of sequence numbers must be increased
- buffering at sender and/or receiver



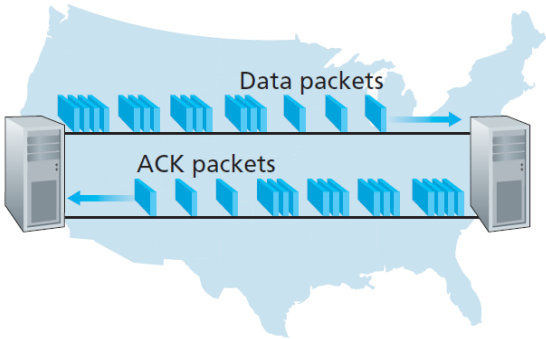
Sliding Window Protocols

- two generic forms of pipelined protocols: *go-Back-N*, *selective repeat*

Non pipelined and pipelined

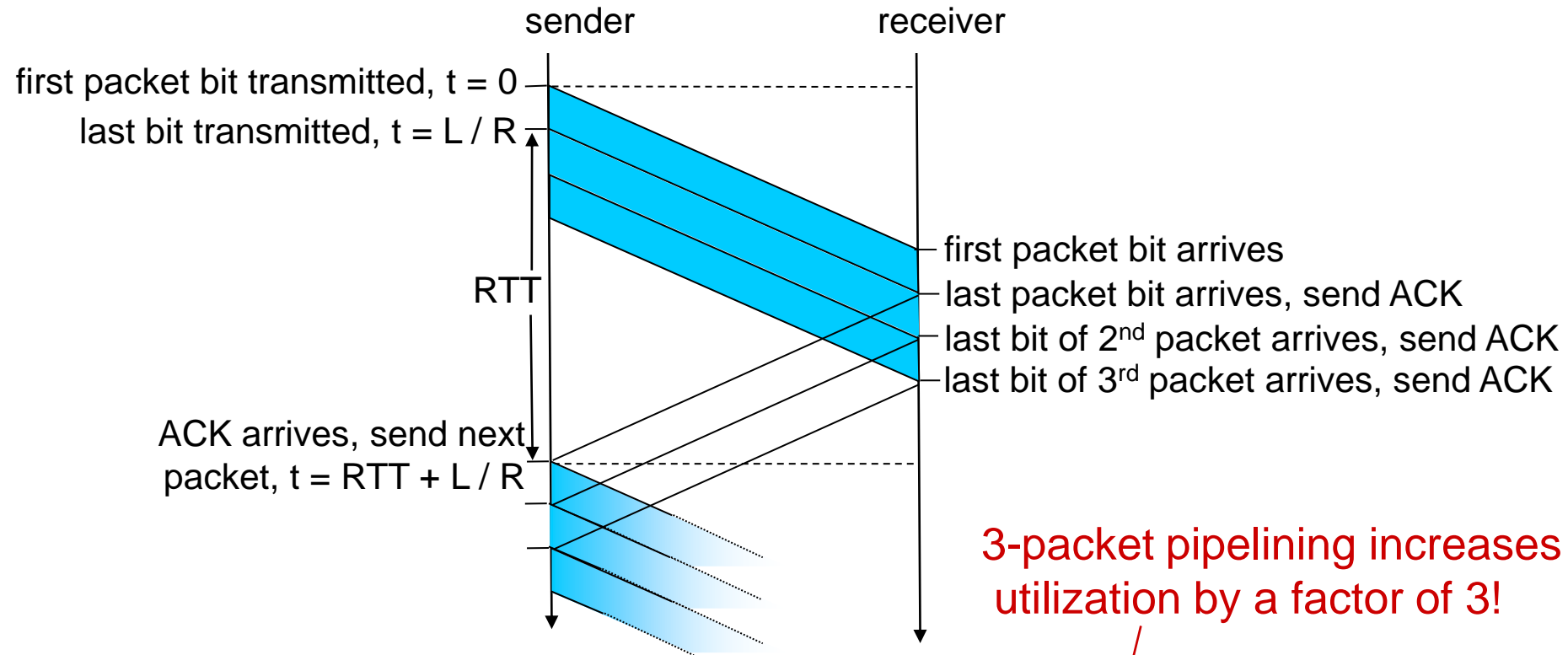


a. A stop-and-wait protocol in operation



b. A pipelined protocol in operation

Pipelining: increased utilization



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

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

Pipelined protocols: **Sliding Window Protocols**

Go-back-N:

- sender can have up to N unacked packets in pipeline
- receiver only sends *cumulative ack*
 - doesn't ack packet if there's a gap
- sender has timer for oldest unacked packet
 - when timer expires, retransmit *all* unacked packets

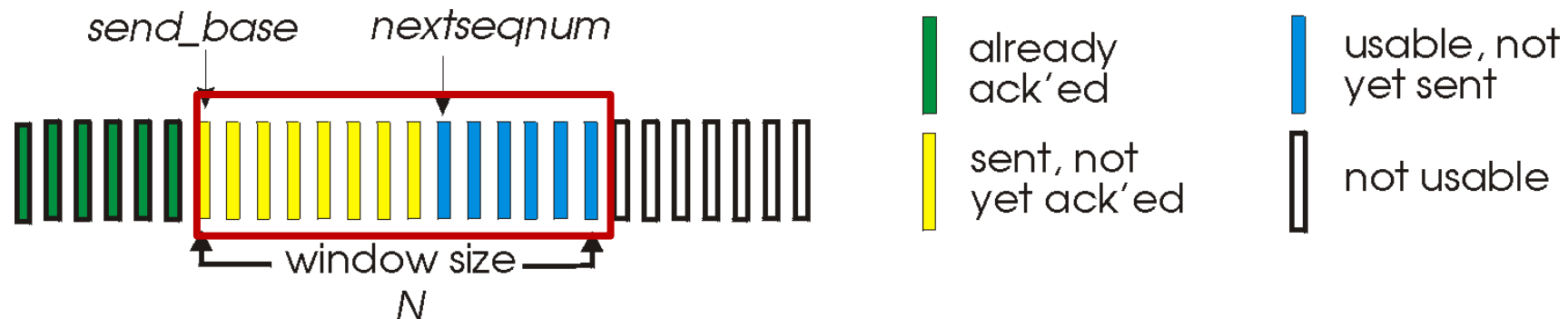
Selective Repeat:

- sender can have up to N unack'ed packets in pipeline
- rcvr sends *individual ack* for each packet
- sender maintains timer for each unacked packet
 - when timer expires, retransmit only that unacked packet

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

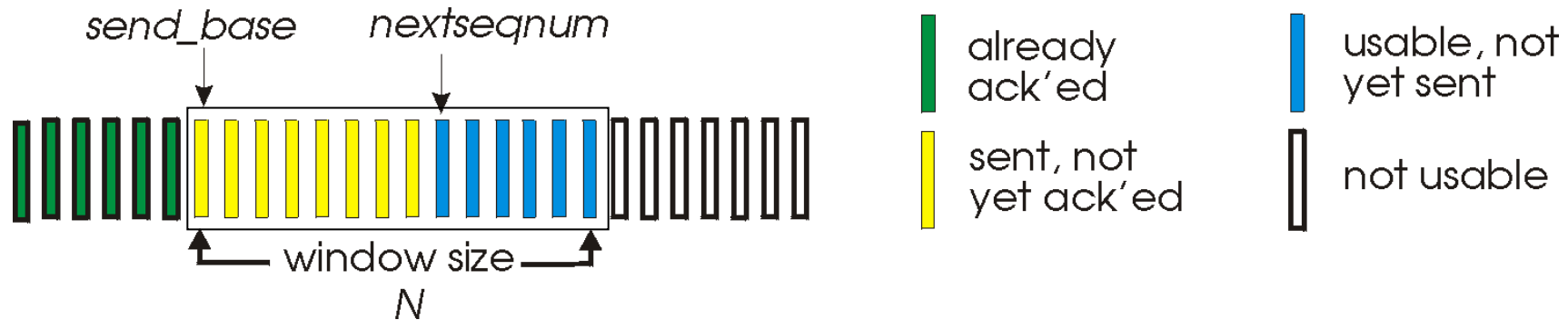
Go-Back-N: sender

- **Base** to be the sequence number of the oldest unacknowledged packet
- **nextseqnum** to be the smallest unused sequence number (that is, the sequence number of the next packet to be sent), then four intervals in the range of sequence numbers can be identified.
- Sequence numbers in the **interval [0, base-1]** correspond to packets that have already been transmitted and acknowledged.
- The **interval [base, nextseqnum-1]** corresponds to packets that have been sent but not yet acknowledged. Sequence numbers in the **interval [nextseqnum, base+N-1]** can be used for packets that can be sent immediately, should data arrive from the upper layer.
- In the end, sequence numbers **greater than or equal to base+N** cannot be used until an unacknowledged packet currently in the pipeline (particularly, the packet with sequence number base) has been acknowledge



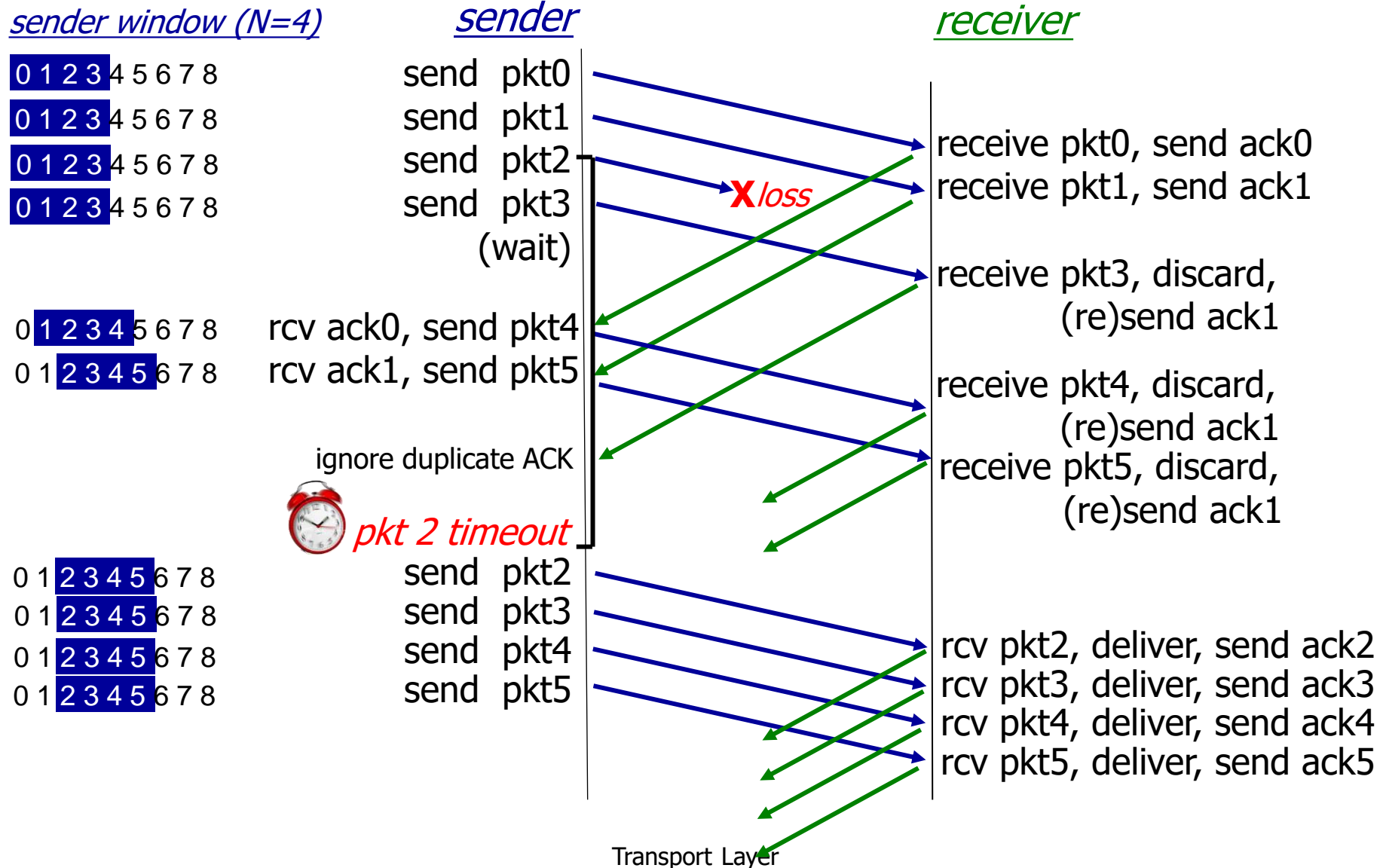
Go-Back-N: sender

- k-bit seq # in pkt header
- “window” of up to N, consecutive unack’ed pkts allowed

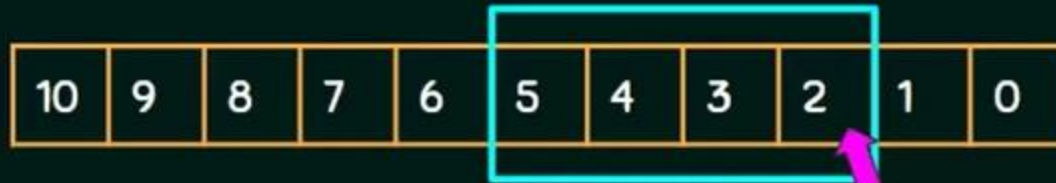


- ACK(n): ACKs all pkts up to, including seq # n - “cumulative ACK”
 - may receive duplicate ACKs (see receiver)
- timer for oldest in-flight pkt
- timeout(n): retransmit packet n and all higher seq # pkts in window

GBN in action

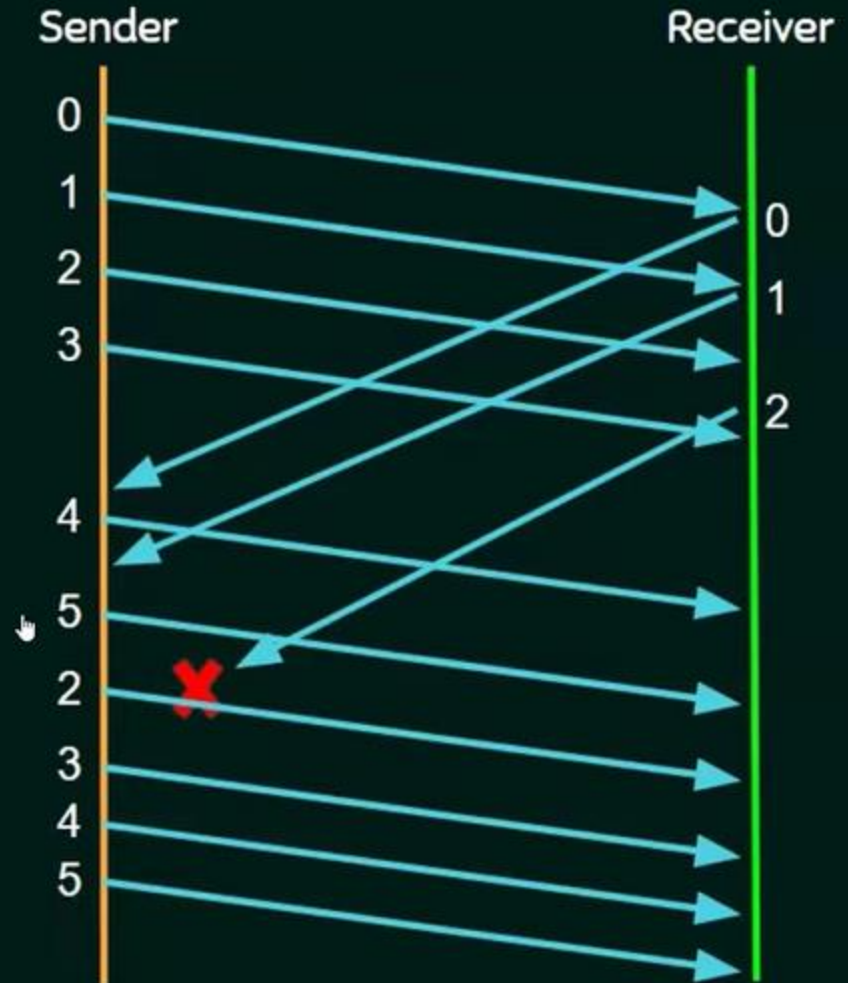


WORKING OF GO-BACK-N ARQ



Window Size:

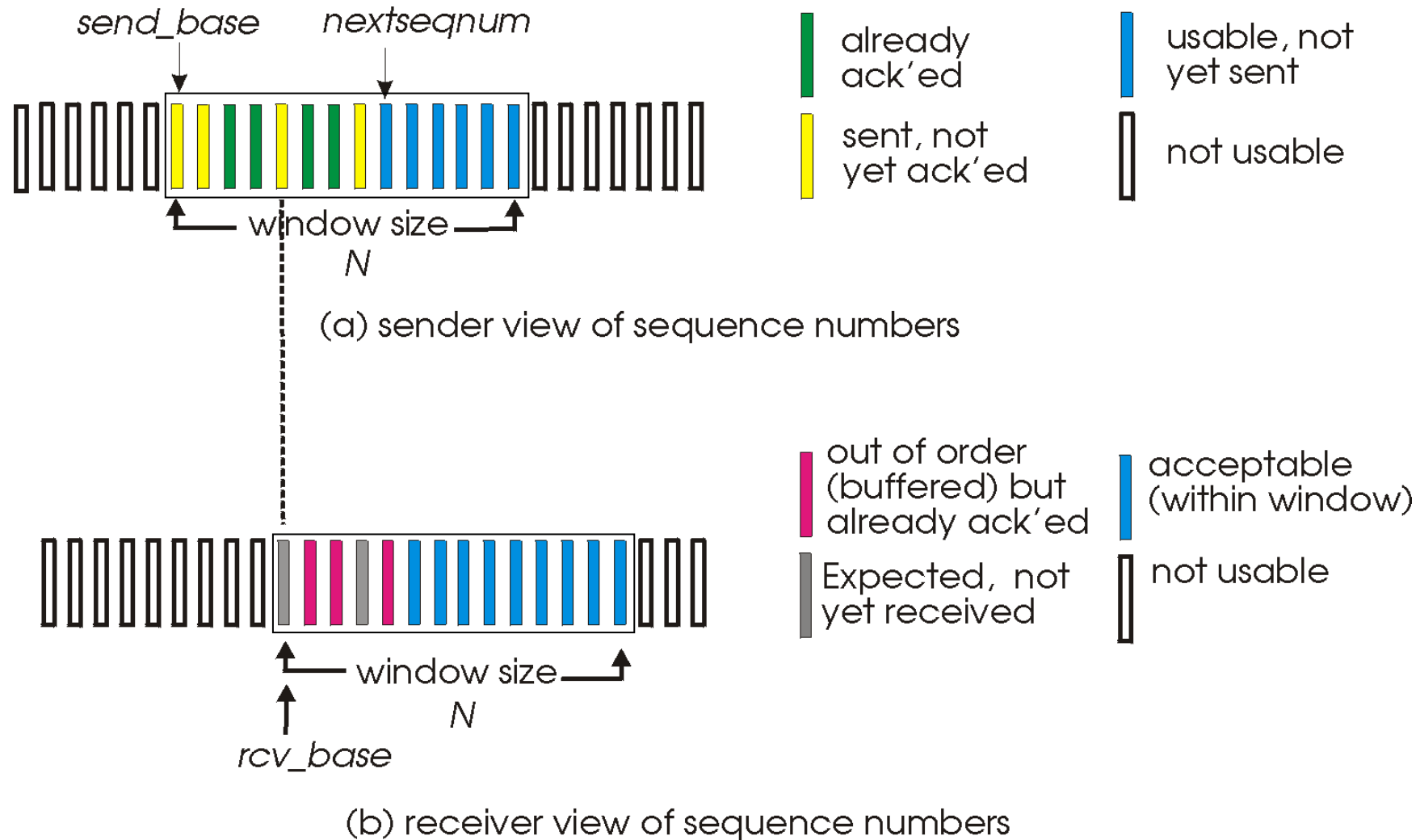
4



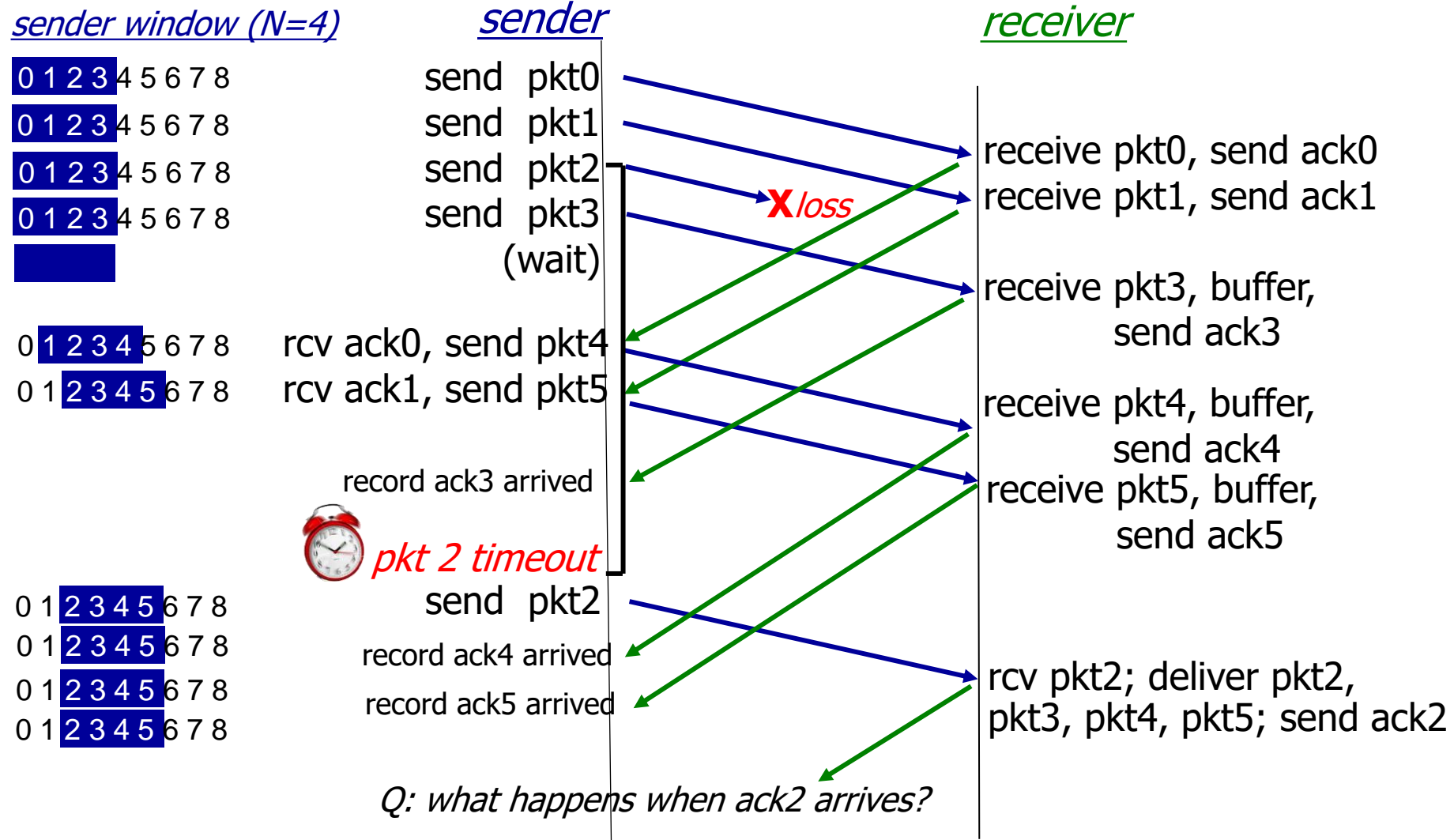
Selective repeat

- receiver *individually* acknowledges all correctly received pkts
 - buffers pkts, as needed, for eventual in-order delivery to upper layer
- sender only resends pkts for which ACK not received
 - sender timer for each unACKed pkt
- sender window
 - N consecutive seq #'s
 - limits seq #'s of sent, unACKed pkts

Selective repeat: sender, receiver windows



Selective repeat in action



Selective repeat

sender

data from above:

- if next available seq # in window, send pkt

timeout(n):

- resend pkt n, restart timer

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

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

receiver

pkt n in [rcvbase, rcvbase+N-1]

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

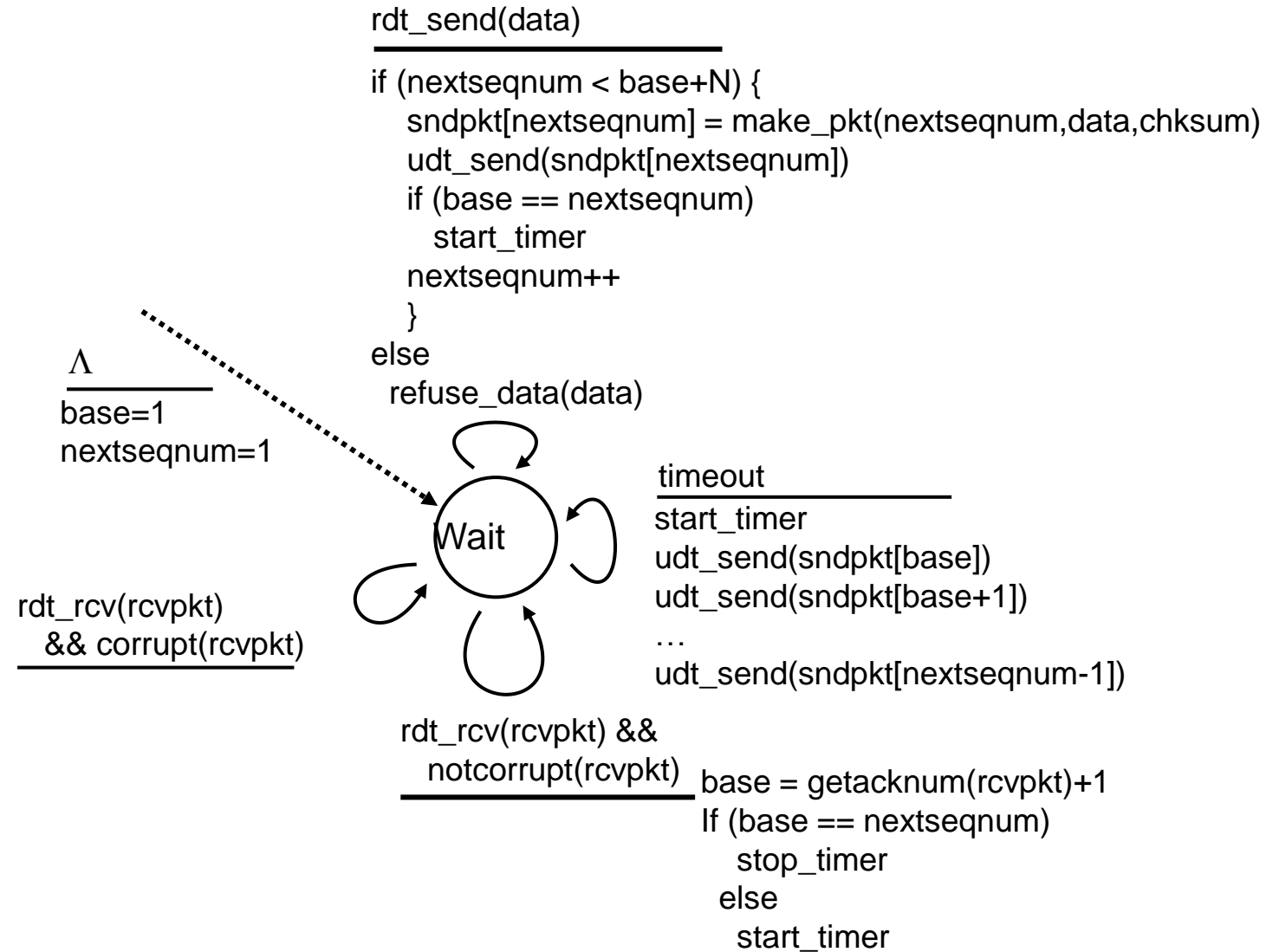
pkt n in [rcvbase-N, rcvbase-1]

- ACK(n)

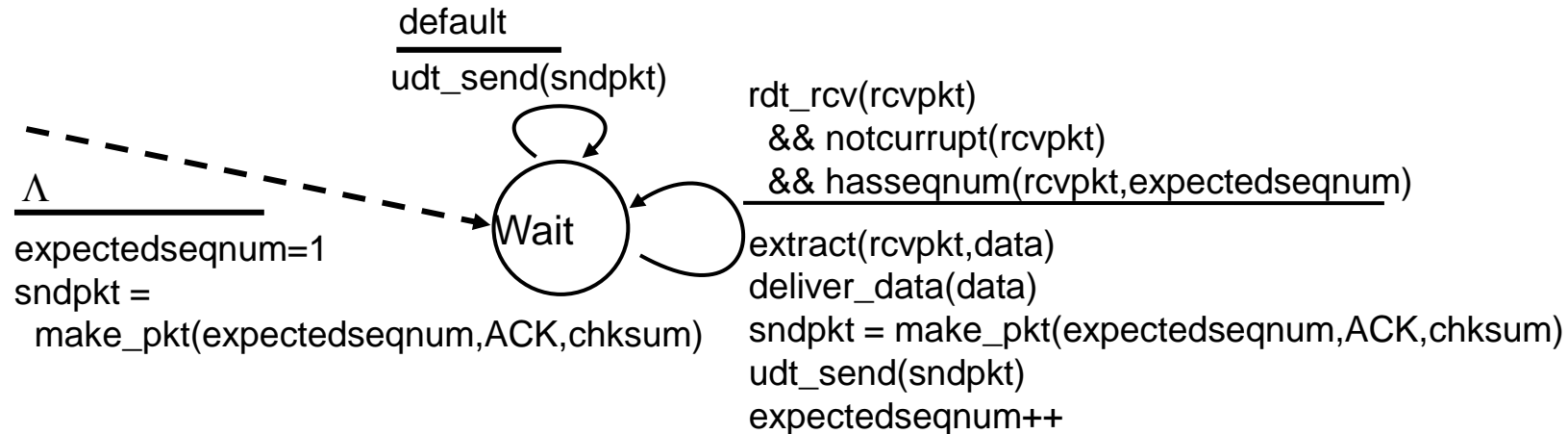
otherwise:

- ignore

GBN: sender extended FSM



GBN: receiver extended FSM



ACK-only: always send ACK for correctly-received pkt with highest *in-order* seq #

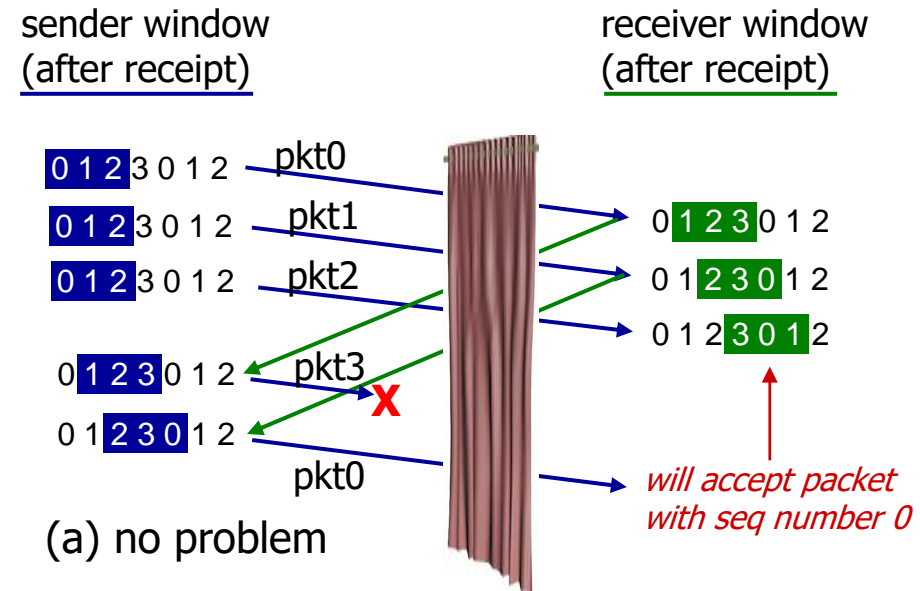
- may generate duplicate ACKs
- need only remember **expectedseqnum**
- out-of-order pkt:
 - discard (don't buffer): *no receiver buffering!*
 - re-ACK pkt with highest in-order seq #

Selective repeat: dilemma

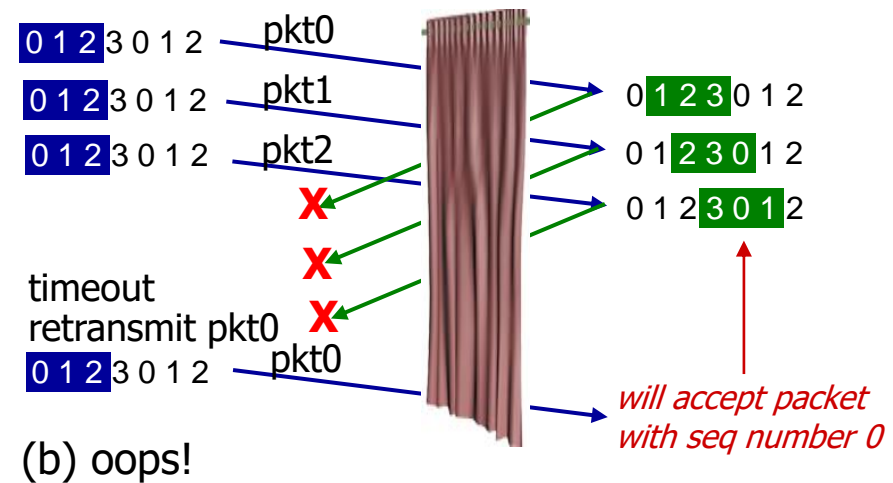
example:

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

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



*receiver can't see sender side.
receiver behavior identical in both cases!
something's (very) wrong!*



Chapter 3 outline

3.1 transport-layer services

3.2 multiplexing and
demultiplexing

3.3 connectionless transport: UDP

3.4 principles of reliable data
transfer

3.5 connection-oriented
transport: TCP

- segment structure
- reliable data transfer
- flow control
- connection management

3.6 principles of congestion
control

3.7 TCP congestion control

Namah Shivaya