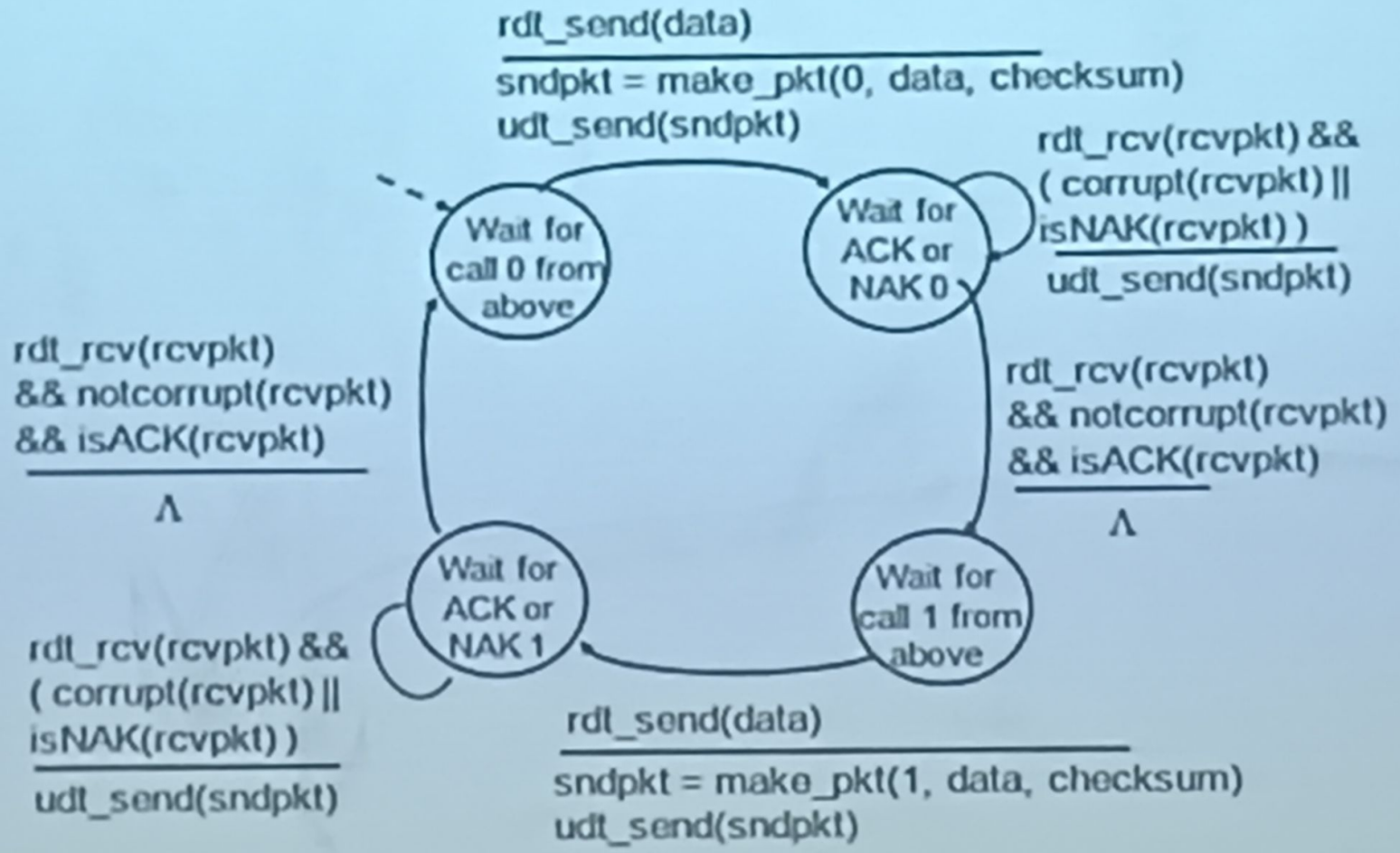
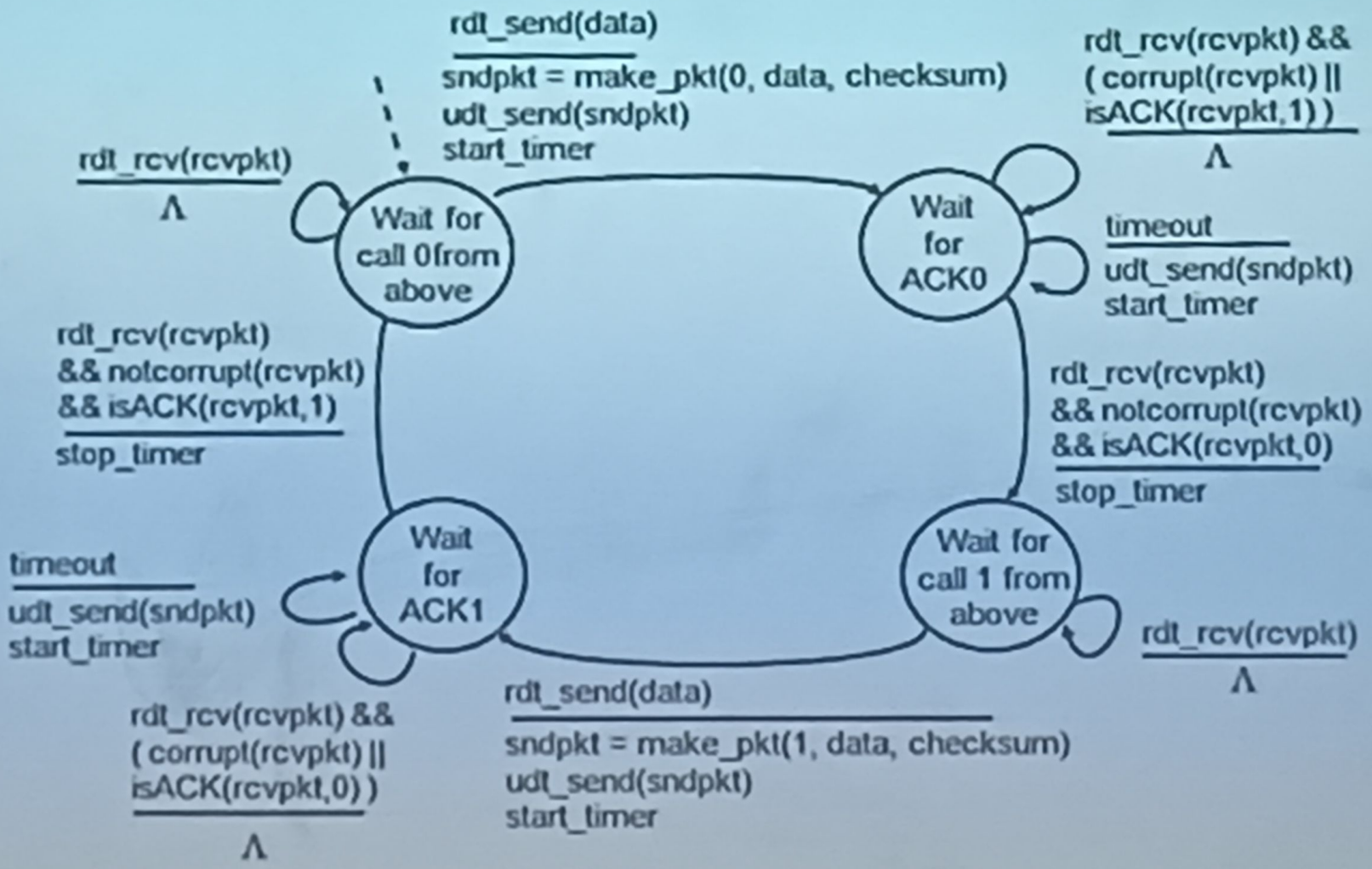
rdt2.1: sender, handles garbled ACK/NAKs



rdt2.2: a NAK-free protocol

- same functionality as rdt2.1, using ACKs only
- instead of NAK, receiver sends ACK for last pkt received OK
  - receiver must *explicitly* include seq # of pkt being ACKed
- duplicate ACK at sender results in same action as NAK: *retransmit current pkt*

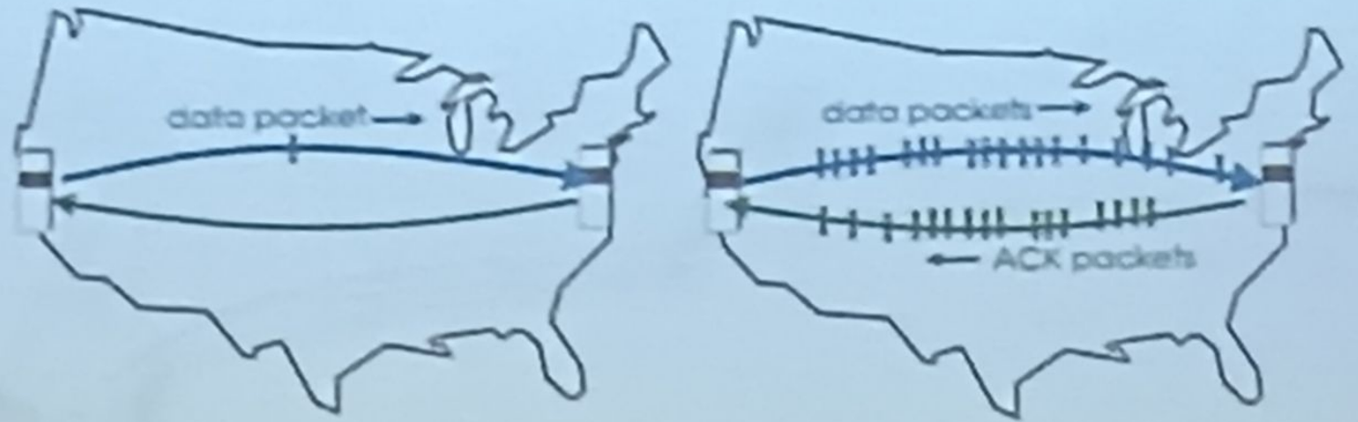
rdt3.0 sender



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



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

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

Pipelining Protocols

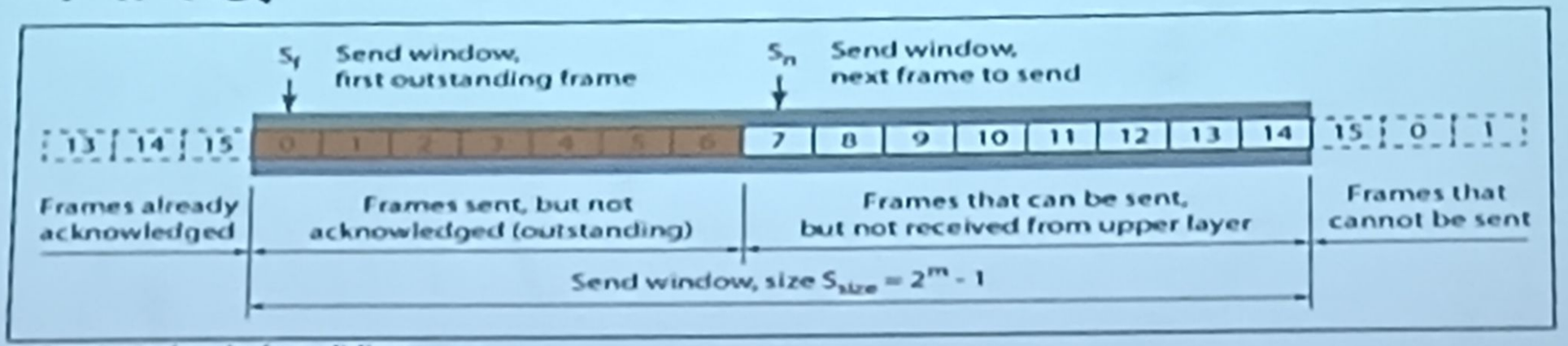
Go-back-N: big picture:

- Sender can have up to N unacked packets in pipeline
- Rcvr only sends cumulative acks
  - Doesn't ack packet if there's a gap or out of order packet received
- Sender has timer for oldest unacked packet
  - If timer expires, retransmit all unacked packets

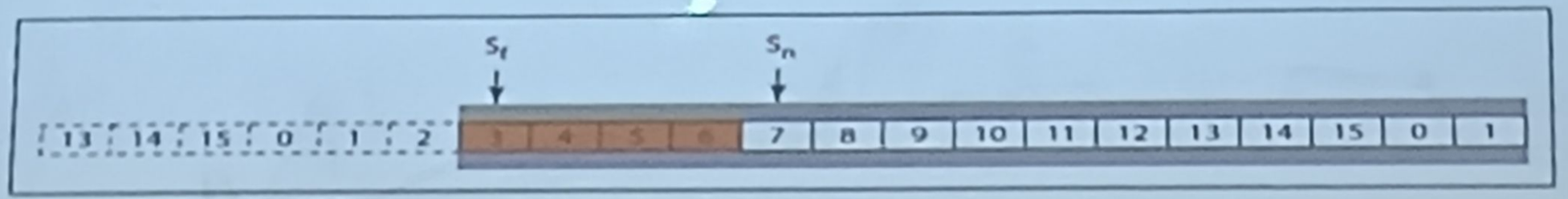
Selective Repeat: big pic

- Sender can have up to N unacked packets in pipeline
- Rcvr acks individual packets
- Sender maintains timer for each unacked packet
  - When timer expires, retransmit only unack packet

Send window for Go-Back-N ARQ



a. Send window before sliding



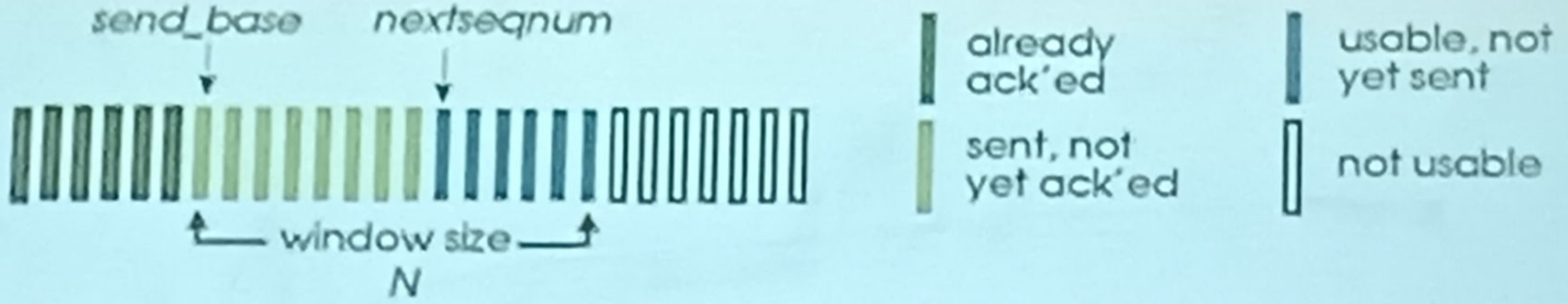
b. Send window after sliding

The send window is an abstract concept defining an imaginary box of size $2^m - 1$ with three variables: S_f, S_n, and S_{size}.
The send window can slide one or more slots when a valid acknowledgment arrives.

Go-Back-N

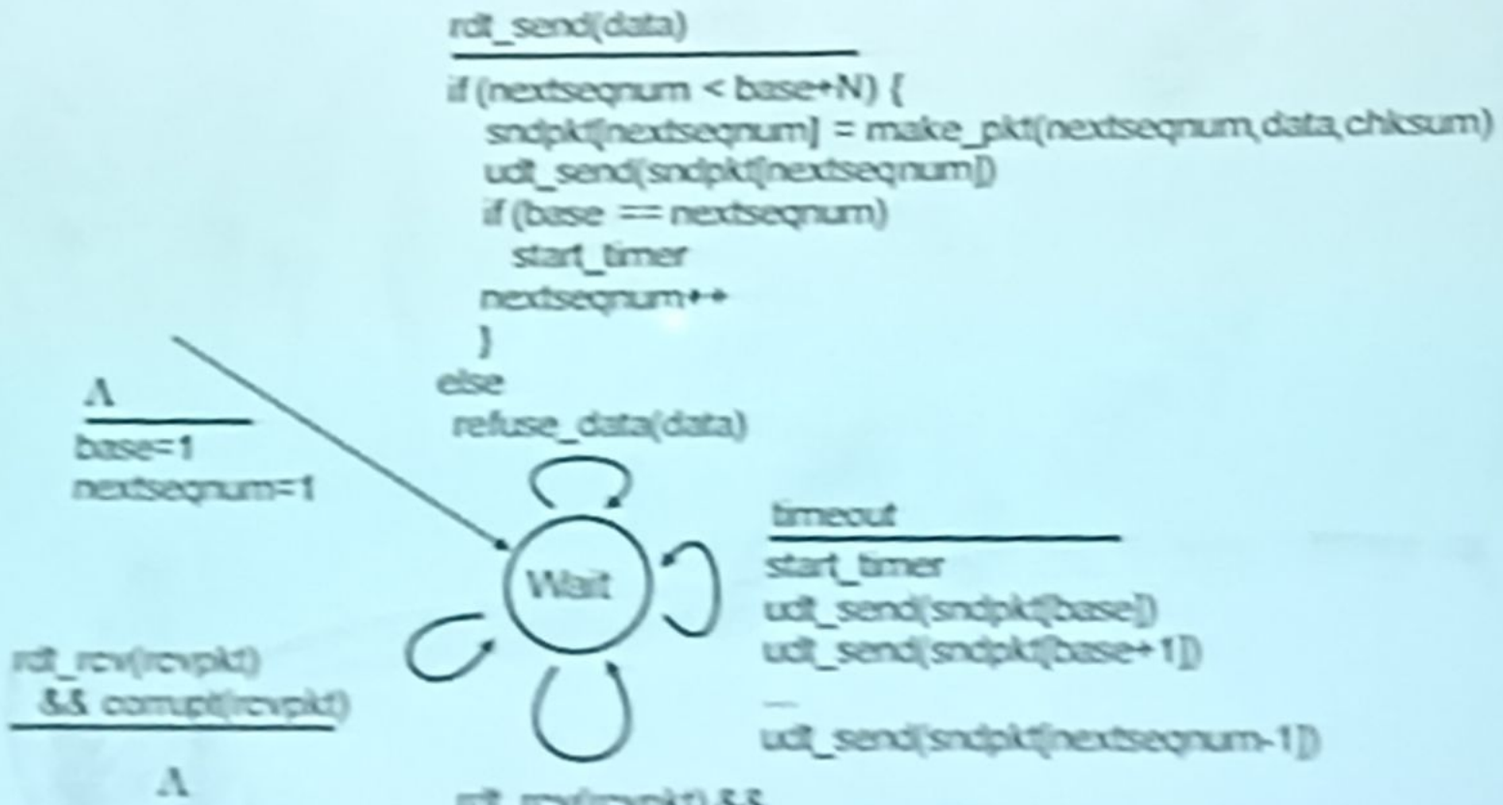
Sender:

- k-bit seq # in pkt header
- "window" of up to N, consecutive unack'ed pkts allowed
- send_base: oldest unacked packet
- nextseqnum: smallest unused seqnum

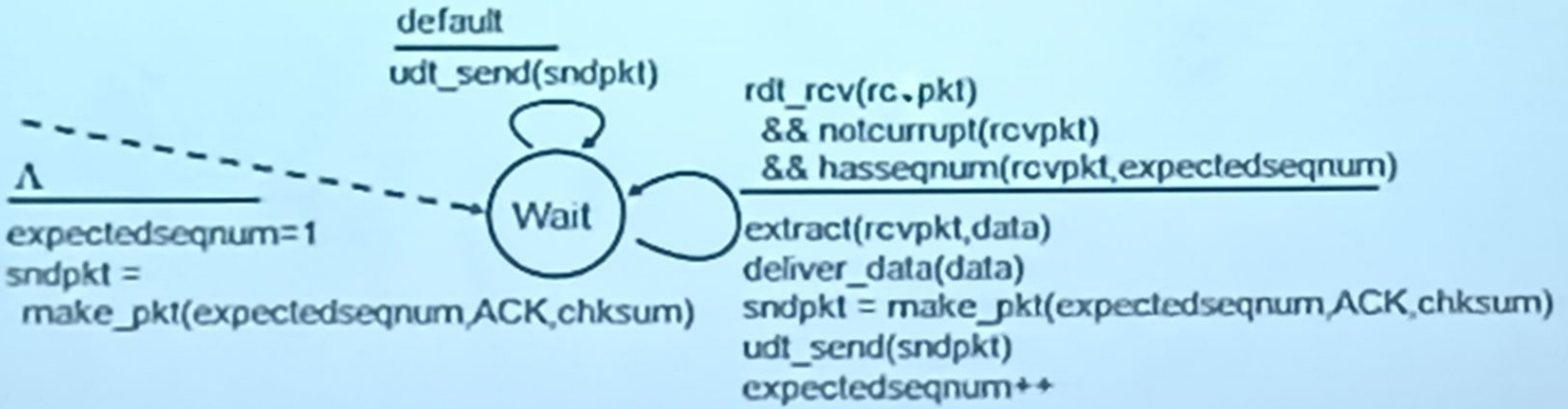


- nextseqnum - send_base ≤ Sender Window Size (SWS=N)
- ACK(n): ACKs all pkts up to, including seq # n - "cumulative ACK"
  - may receive duplicate ACKs (see receiver)
- timer for each in-flight pkt
- *timeout(n):* retransmit pkt n and all higher seq # pkts in window

GBN: sender extended FSM



GBN: receiver extended FSM

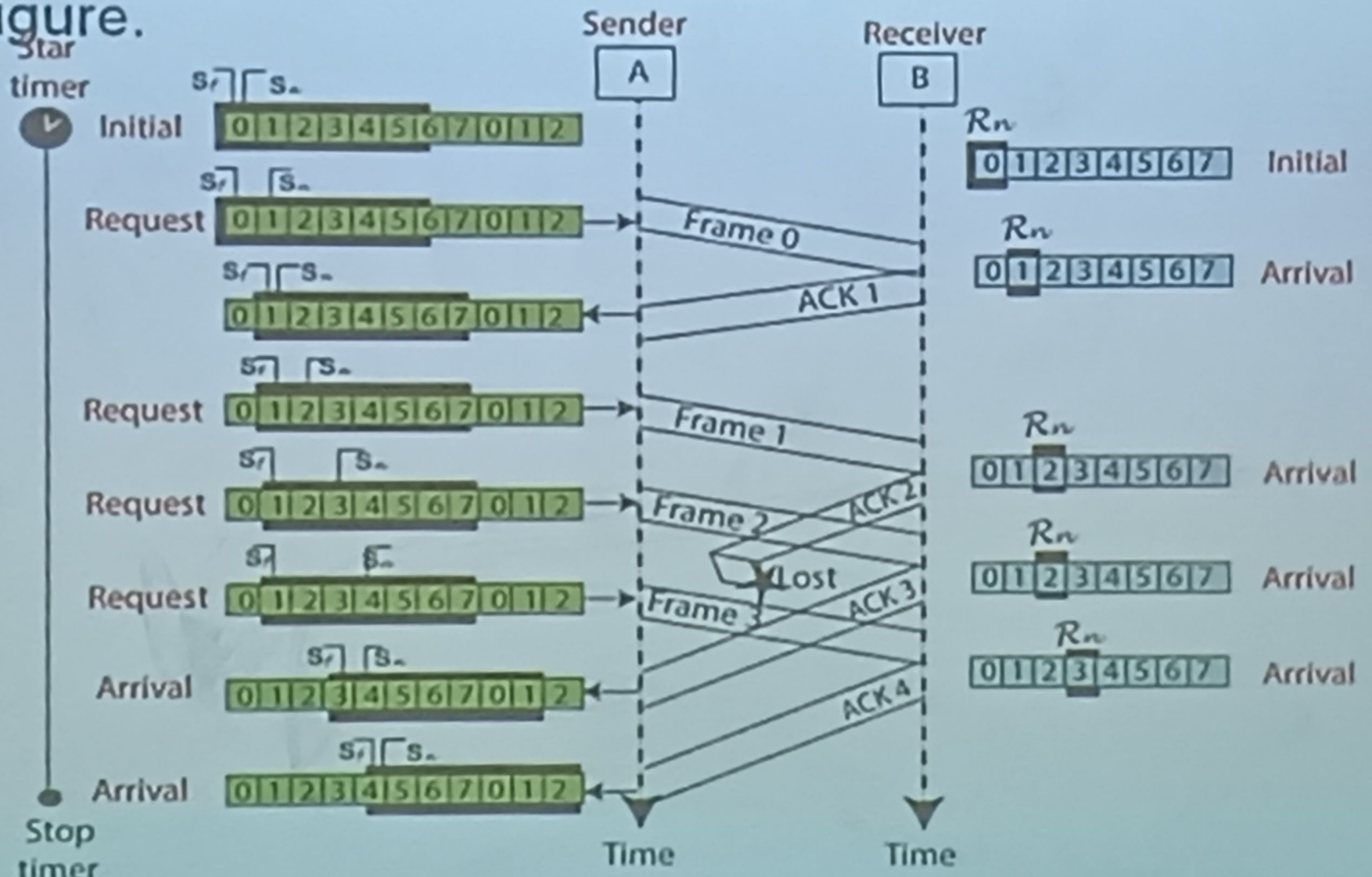


ACK-only: always send ACK for correctly-received pkt with highest *in-order* seq #

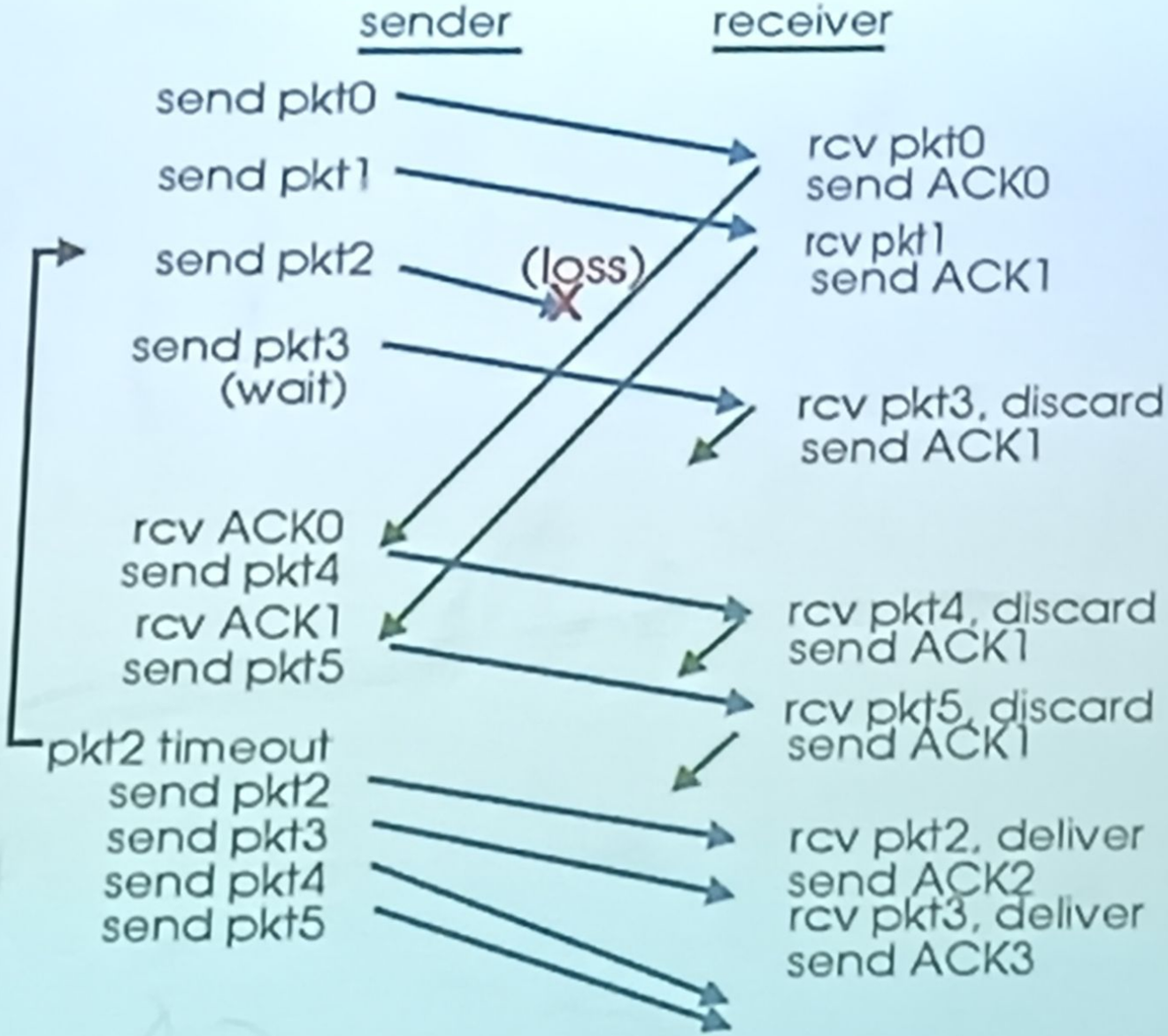
- may generate duplicate ACKs
- need only remember **`expectedseqnum`**

- out-of-order pkt:
  - discard (don't buffer) -> no receiver buffering!
    - Is it bad?
  - Re-ACK pkt with highest in-order seq #

The example of Go-Back-N is shown below in the figure.

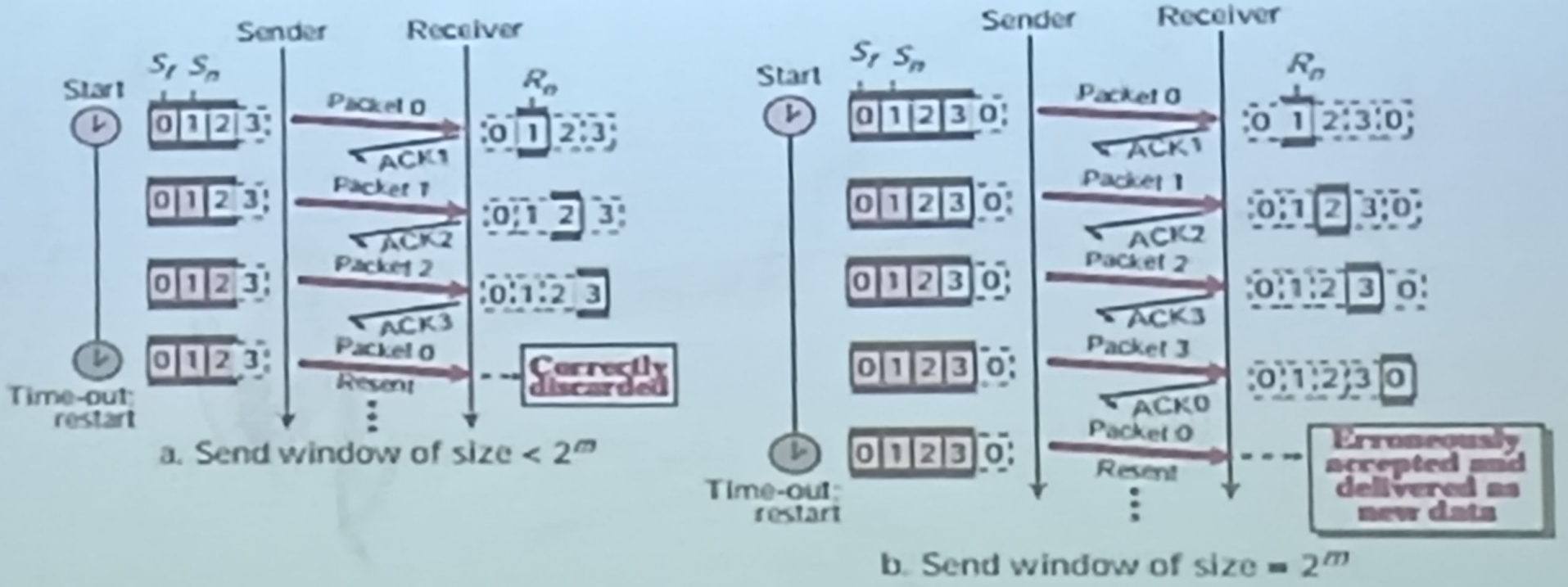


GBN in action



Sender Window size

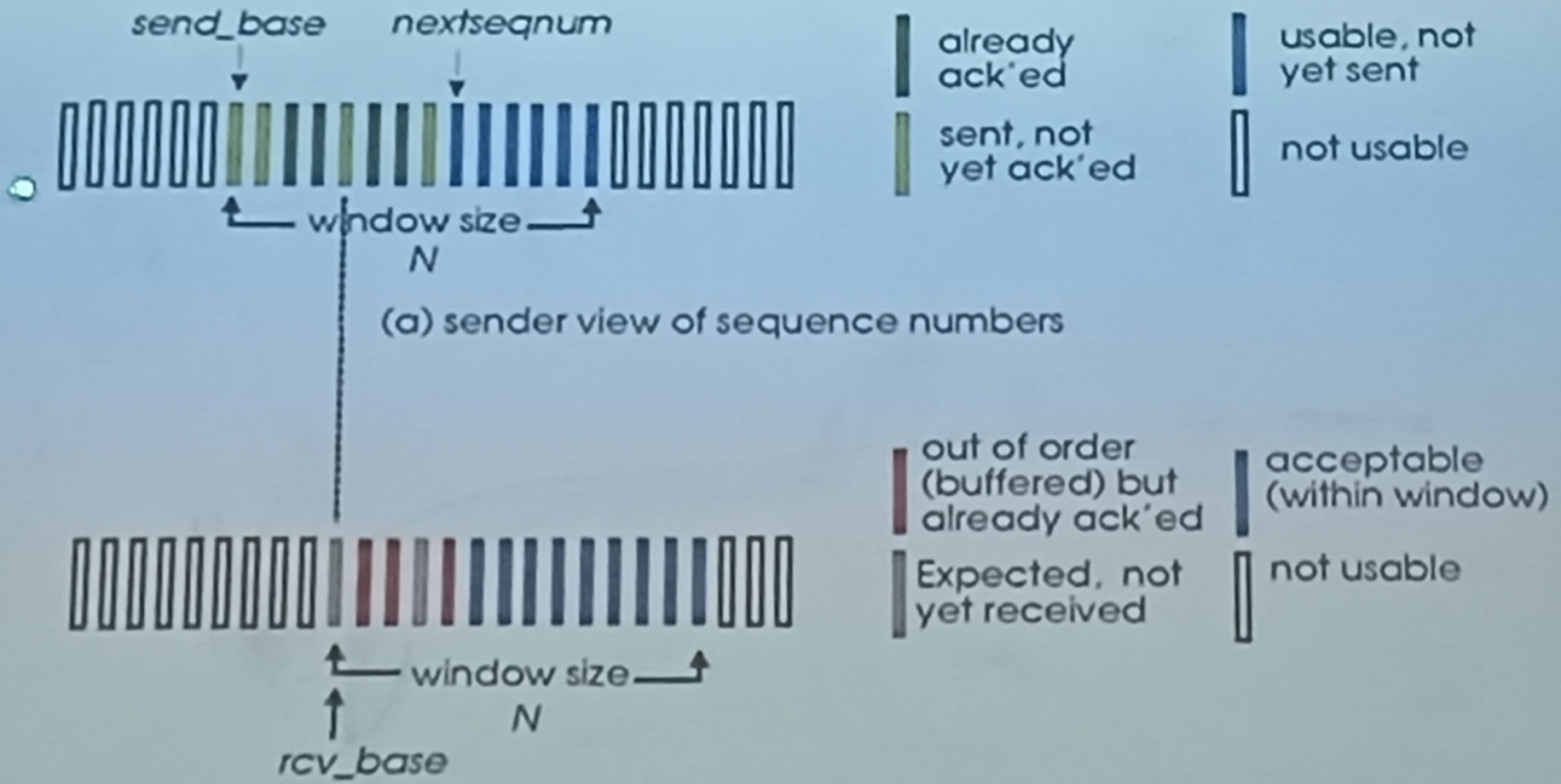
Send window size for Go-Back-N



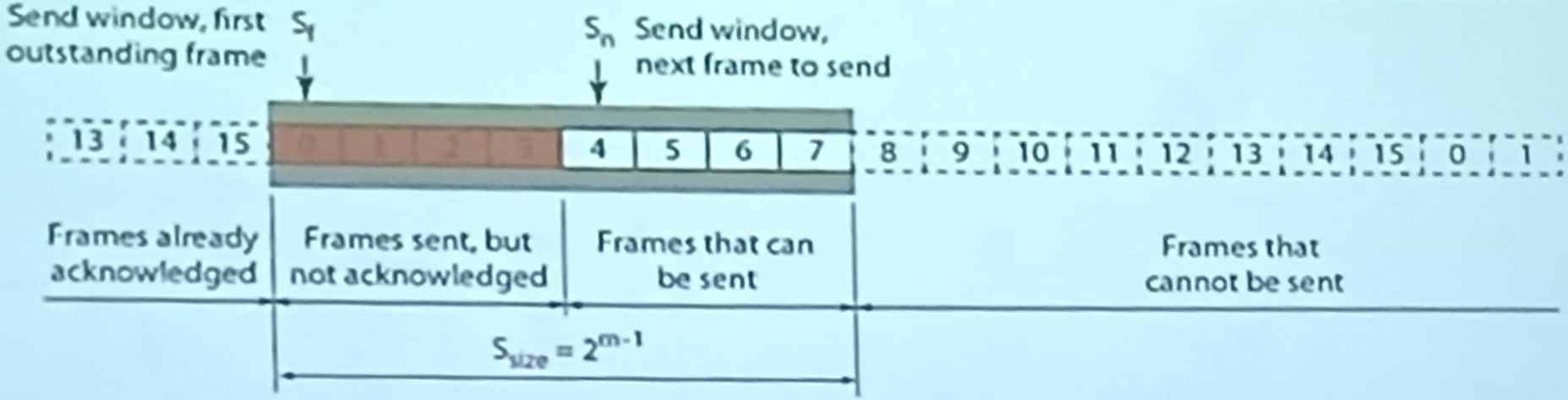
Selective repeat: big picture

- Sender can have up to N unacked packets in pipeline
- Rcvr acks individual packets
- Sender maintains timer for each unacked packet
  - When timer expires, retransmit only unack packet

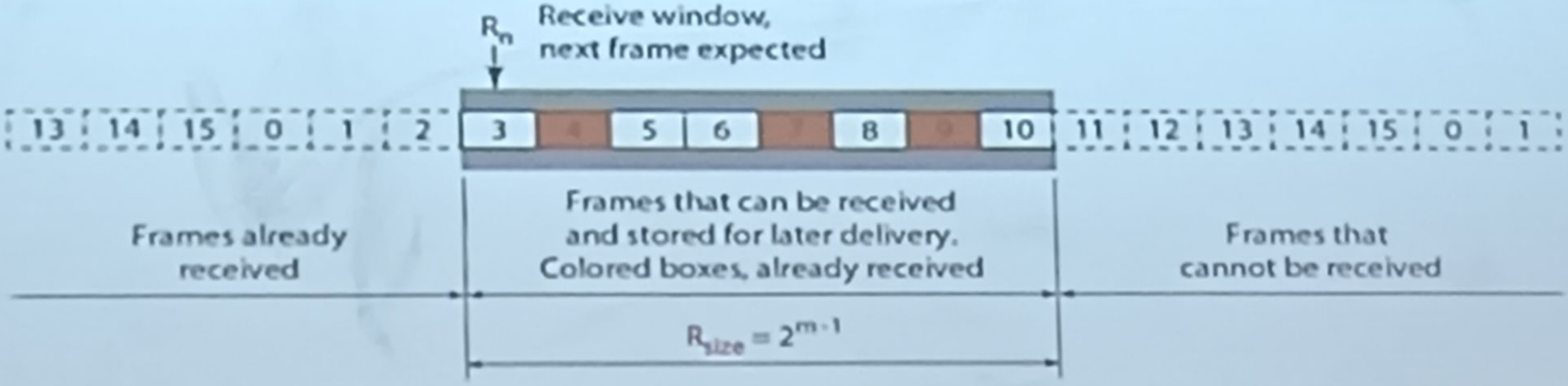
Selective repeat: sender, receiver windows



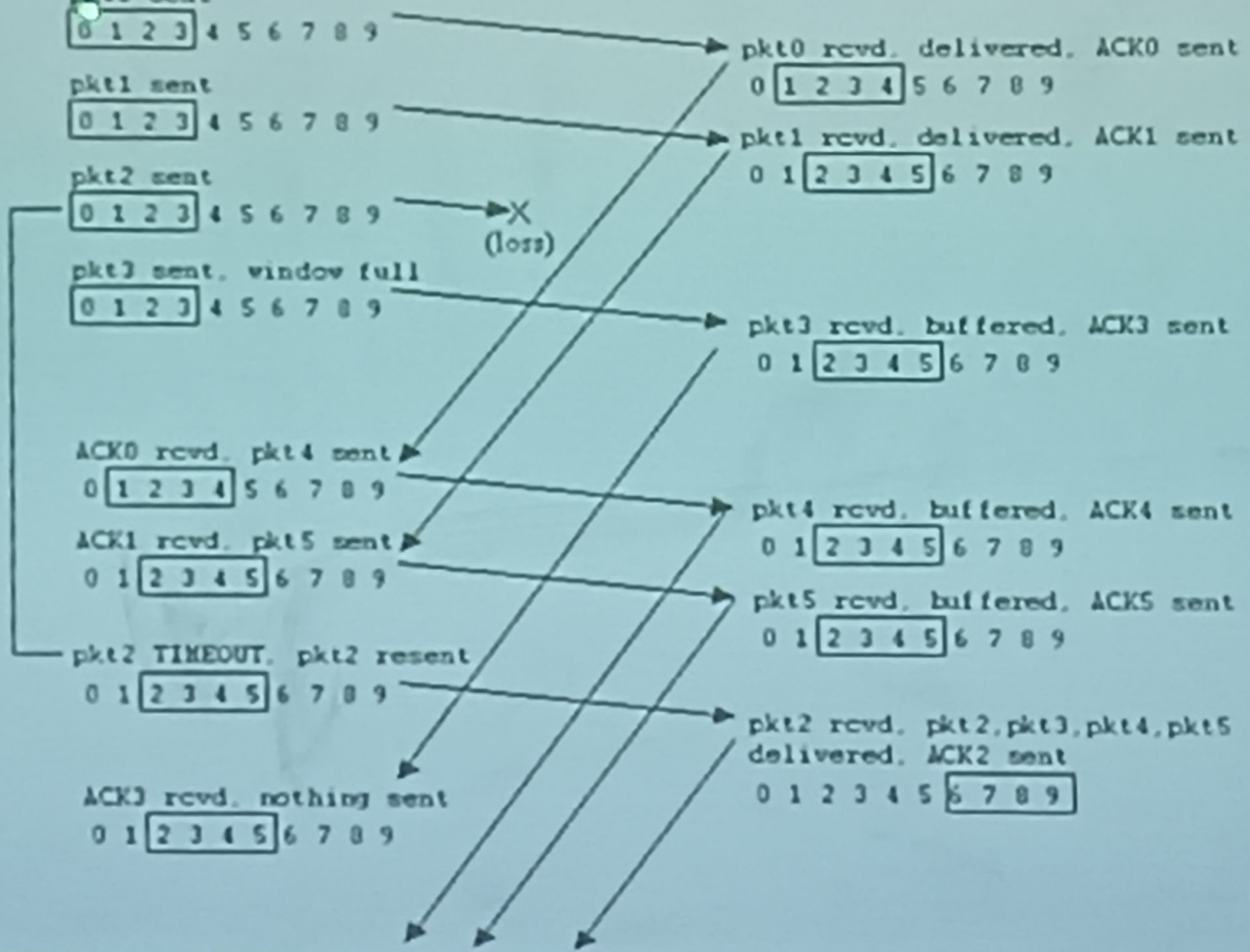
Send window for Selective Repeat ARQ



Receive window for Selective Repeat ARQ



Selective repeat in action



Selective repeat: dilemma

Example:

- seq #'s: 0, 1, 2, 3
- window size=3

- receiver sees no difference in two scenarios!
- incorrectly passes duplicate data as new in (a)

Q: what relationship between seq # size and window size?