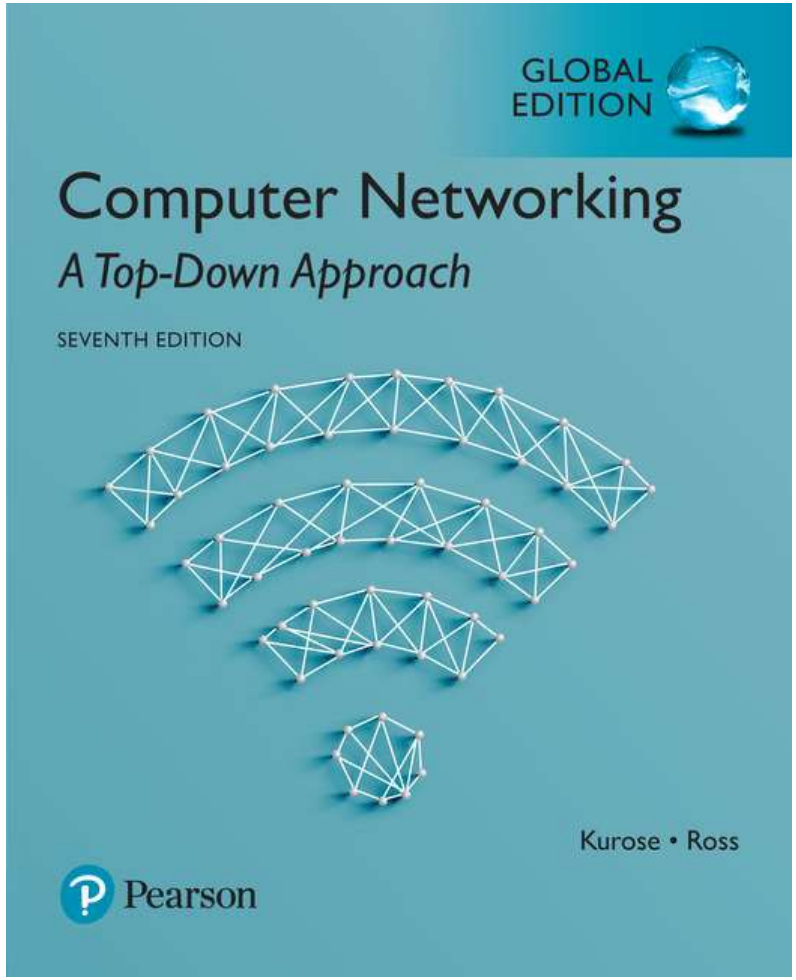
Chapter 3
제13강 Go-back-N & Selective Repeat

Computer Networking: A Top Down Approach

컴퓨터 네트워크
(2019년 1학기)

박승철교수

한국기술교육대학교
컴퓨터공학부



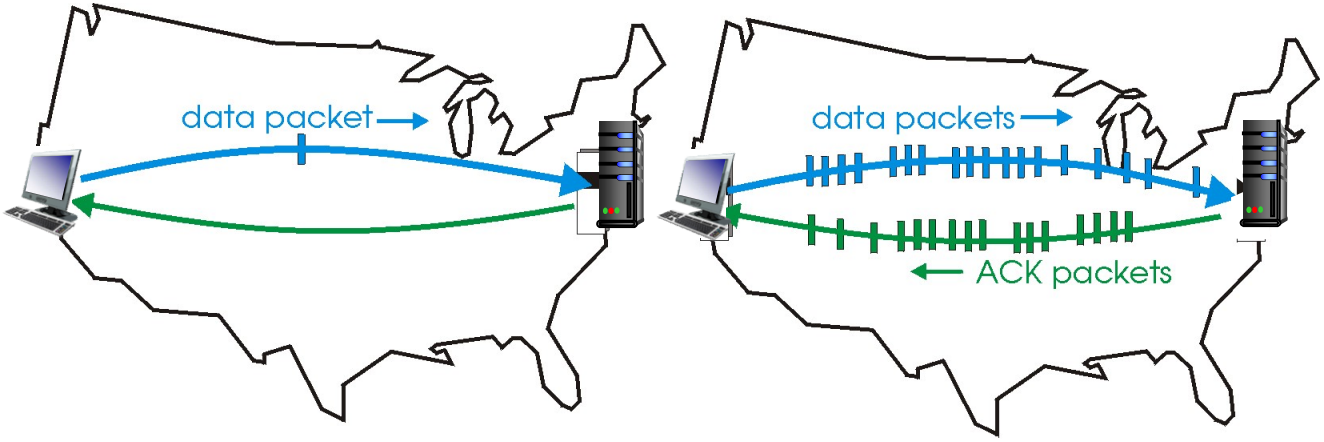
Pre-study Test :

- 순서번호가 가질 수 있을 값의 범위는 [0, 255]이다.
  - ① 순서번호 필드의 크기는 몇 비트인가?
  - ② GBN의 최대 송신 윈도우 크기는 얼마인가?
  - ③ 선택적 반복에서 최대 송신 윈도우 크기는 얼마인가?

Pipelined protocols

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

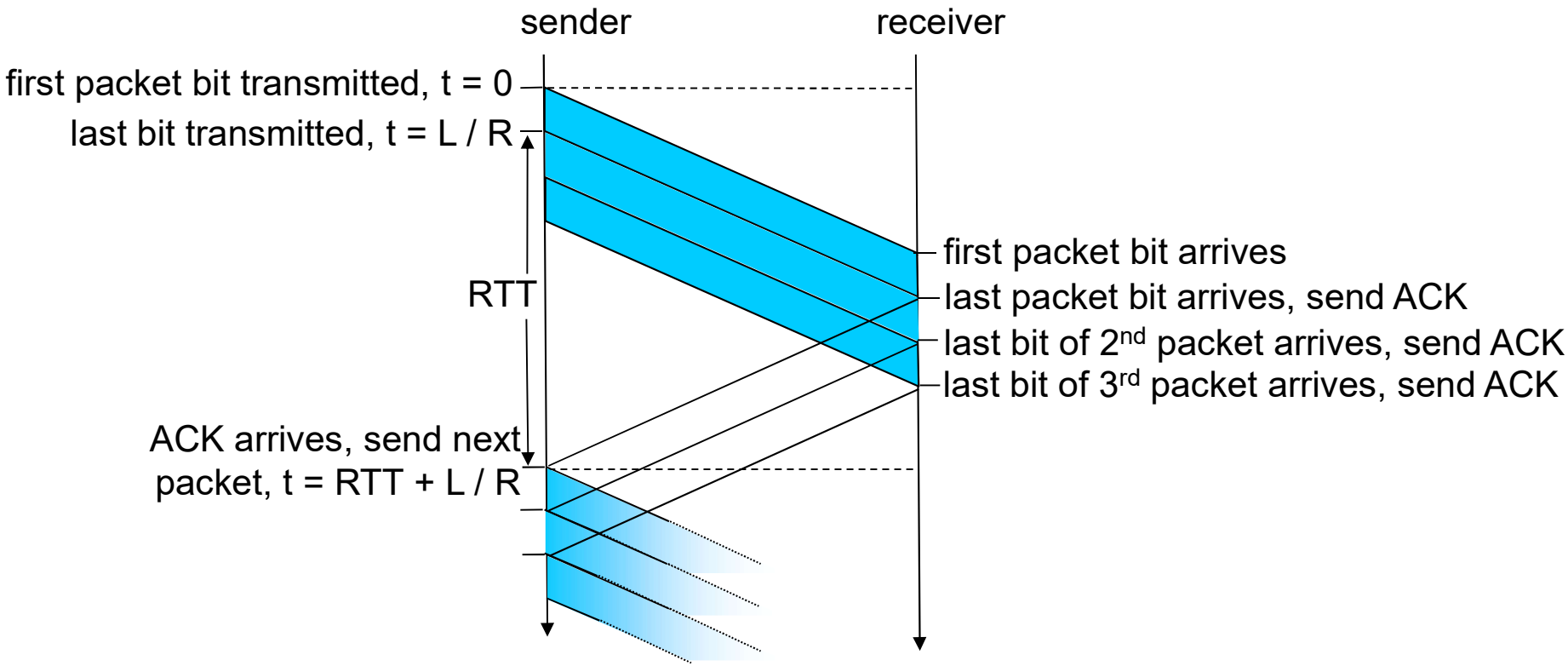
- range of sequence numbers must be increased
- buffering at sender and/or receiver



(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: increased utilization



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

Pipelined protocols: overview

Go-back-N:

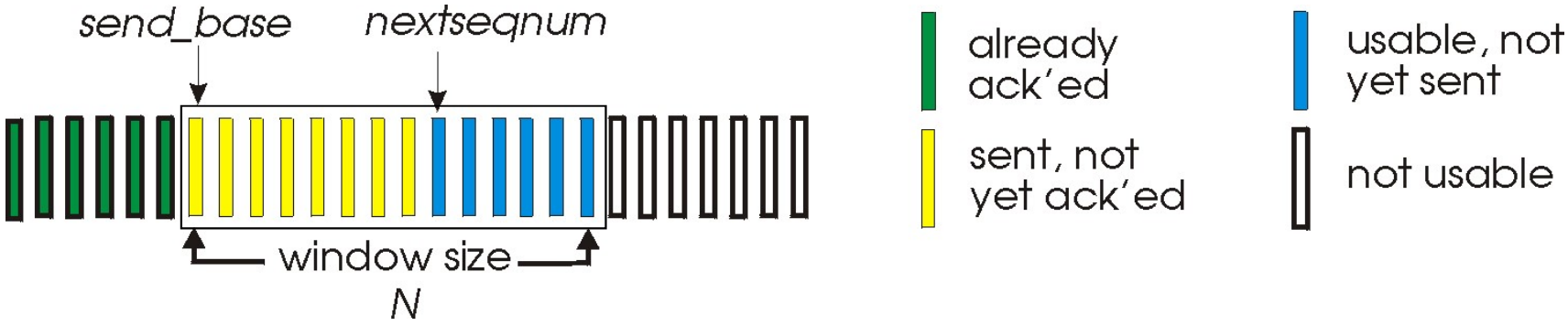
- sender can have up to N unacked packets in pipeline
- receiver only sends *cumulative ack(누적 수신 확인)*
  - doesn't ack packet if there's a gap
- sender has timer for oldest unacked packet
  - when timer expires, retransmit *all* unacked packets

Selective Repeat:

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

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"(누적 수신 확인)*
  - may receive duplicate ACKs (see receiver)
- timer for oldest in-flight pkt(가장 오래된 전송중 패킷에 대한 타이머)
- *timeout(n):* retransmit packet n and all higher seq # pkts in window

Go-Back-N: sender

- 순서번호(sequence number)
  - 패킷 헤더 필드에 표시
  - 제한된 범위의 값(예, TCP - 32 비트)
- 윈도우(window)
  - 확인응답(Ack) 없이 전송할 수 있는 패킷의 순서 번호 범위
  - 확인응답 수신시 이동(sliding window)
  - 윈도우 크기(고정 간주) : 수신 버퍼 등에 의해 결정

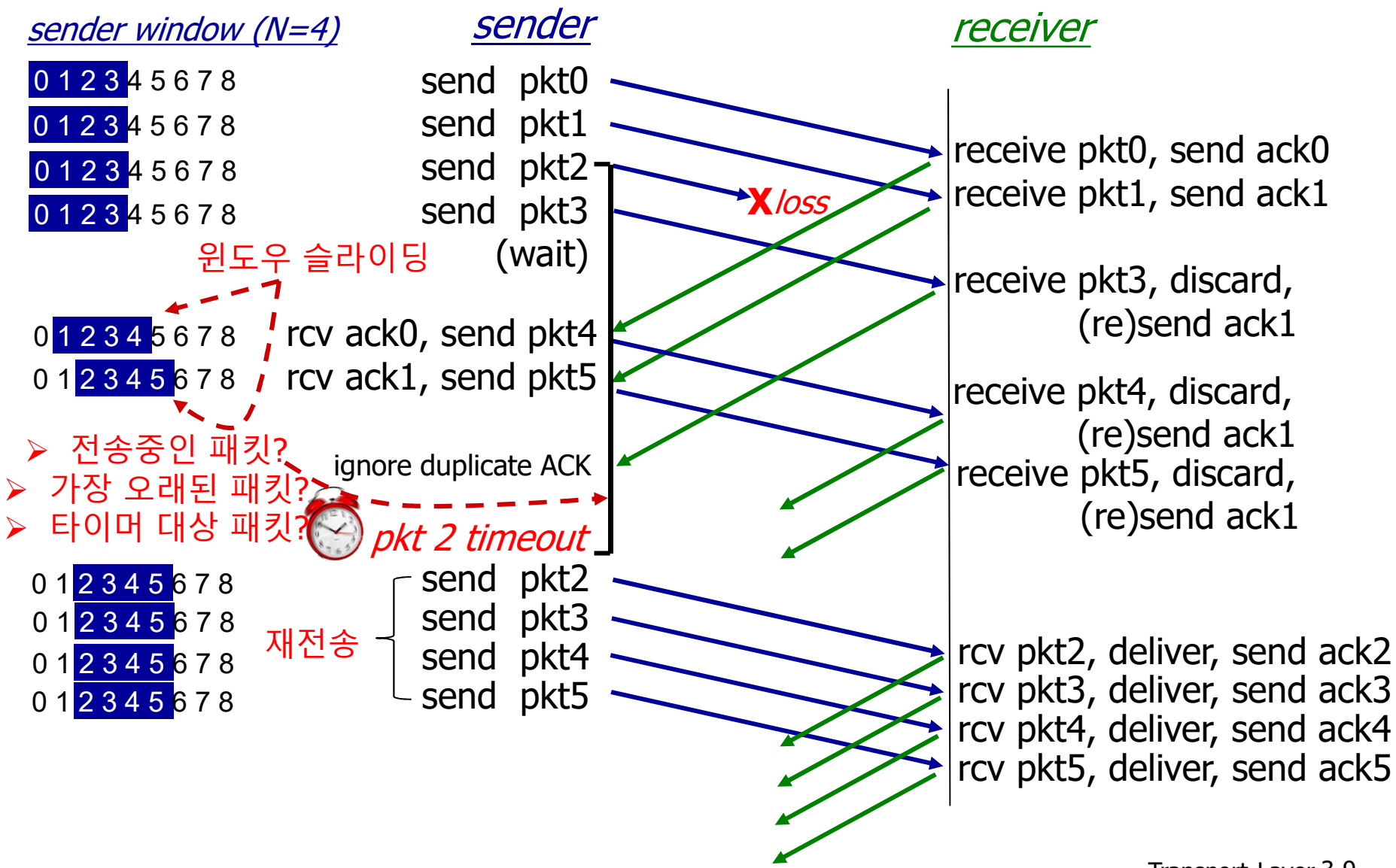
GBN: receiver extended FSM

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 #

GBN in action



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

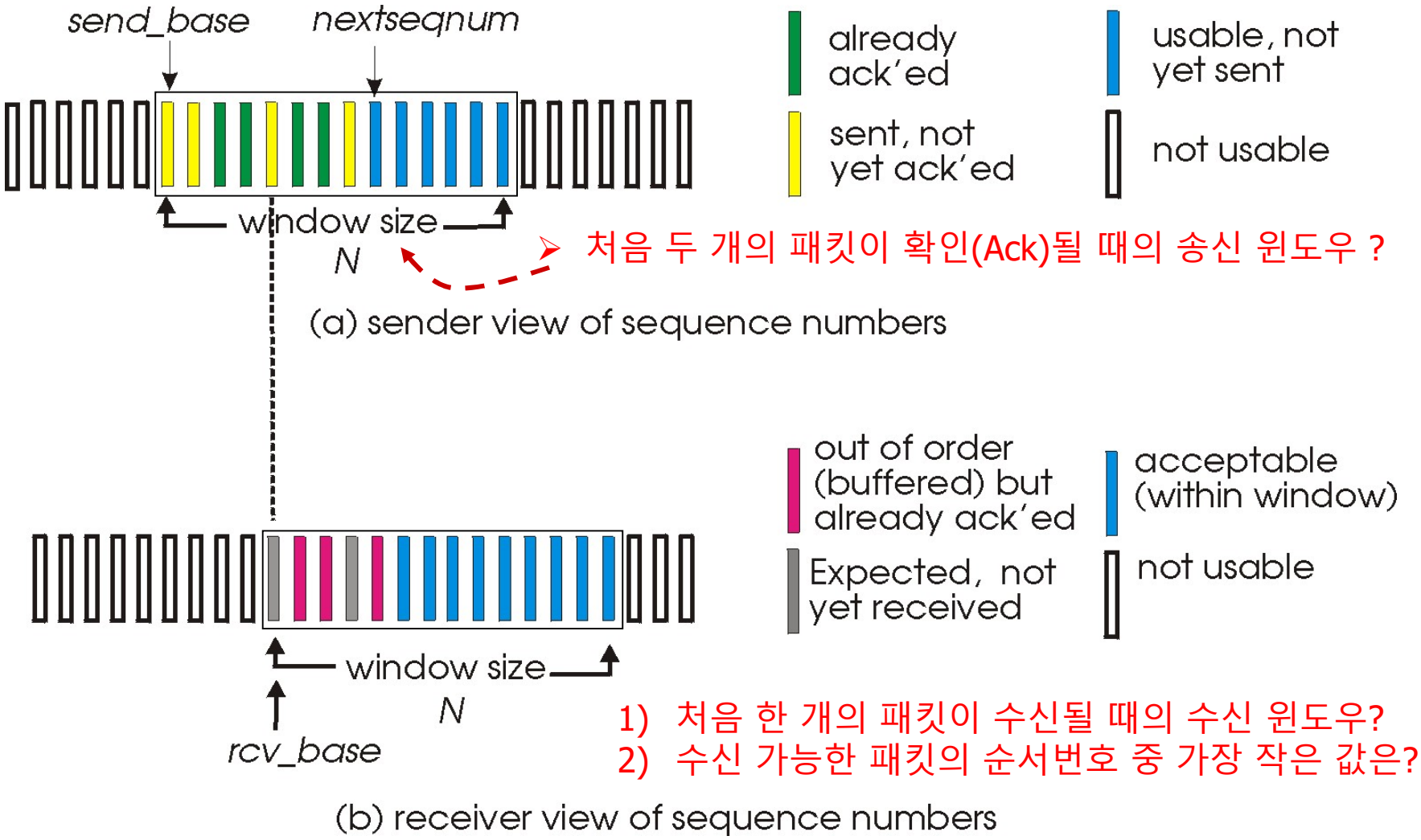
Selective repeat

- 송신 윈도우(sending window)
  - 확인응답(Ack) 없이 전송할 수 있는 패킷의 순서 번호 범위
  - 윈도우 크기 : 고정 간주
  - ➢윈도우 이동 시점 ?

Selective repeat

- 수신 윈도우(receiving window)
  - 수신자가 수신할 수 있어야 하는(저장해야 하는) 패킷의 순서번호 범위
  - ➢윈도우 이동 시점 ?

Selective repeat: sender, receiver windows



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

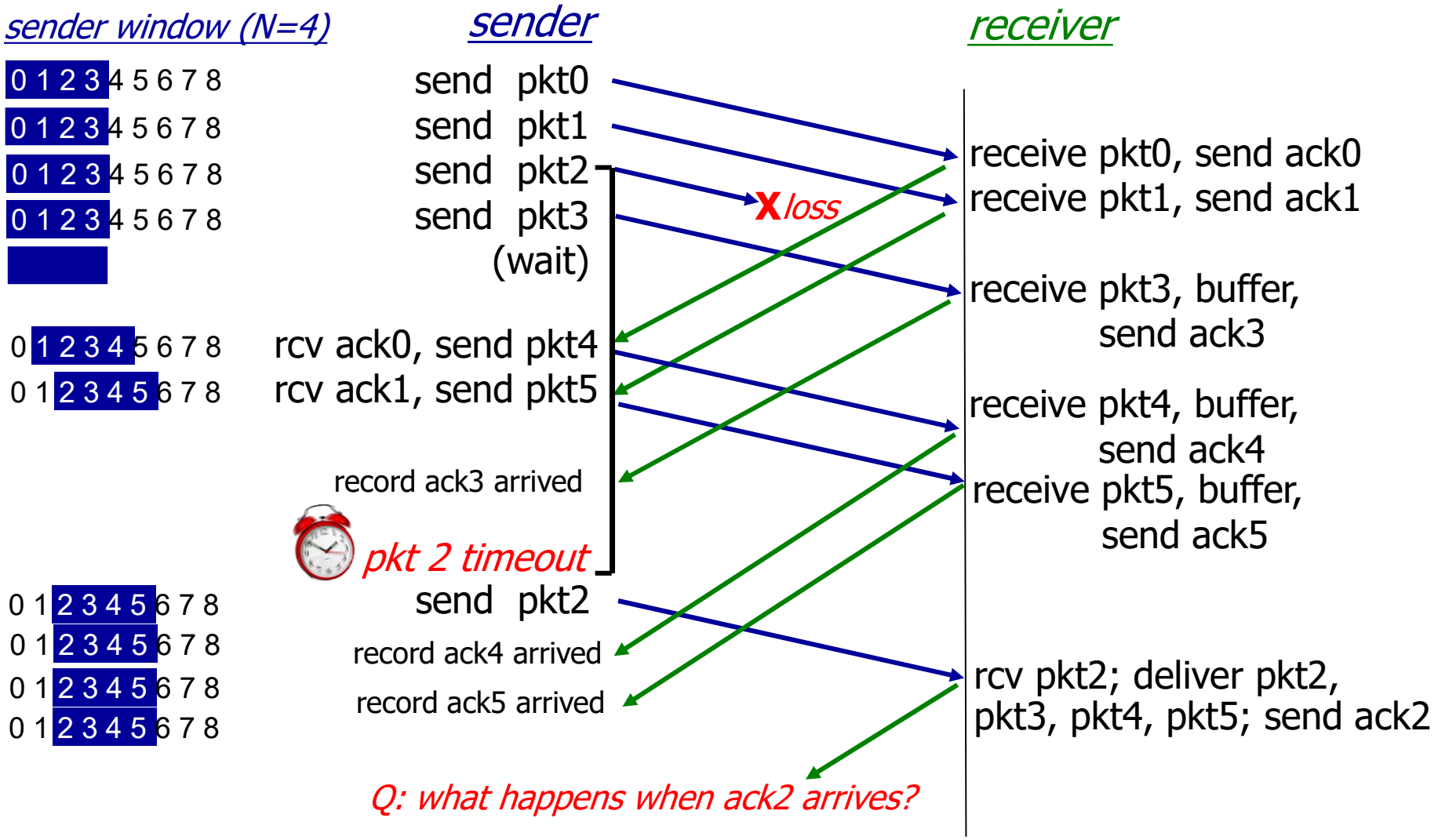
pkt n in [rcvbase-N,rcvbase-1]

- ACK(n)

otherwise:

- ignore

Selective repeat in action

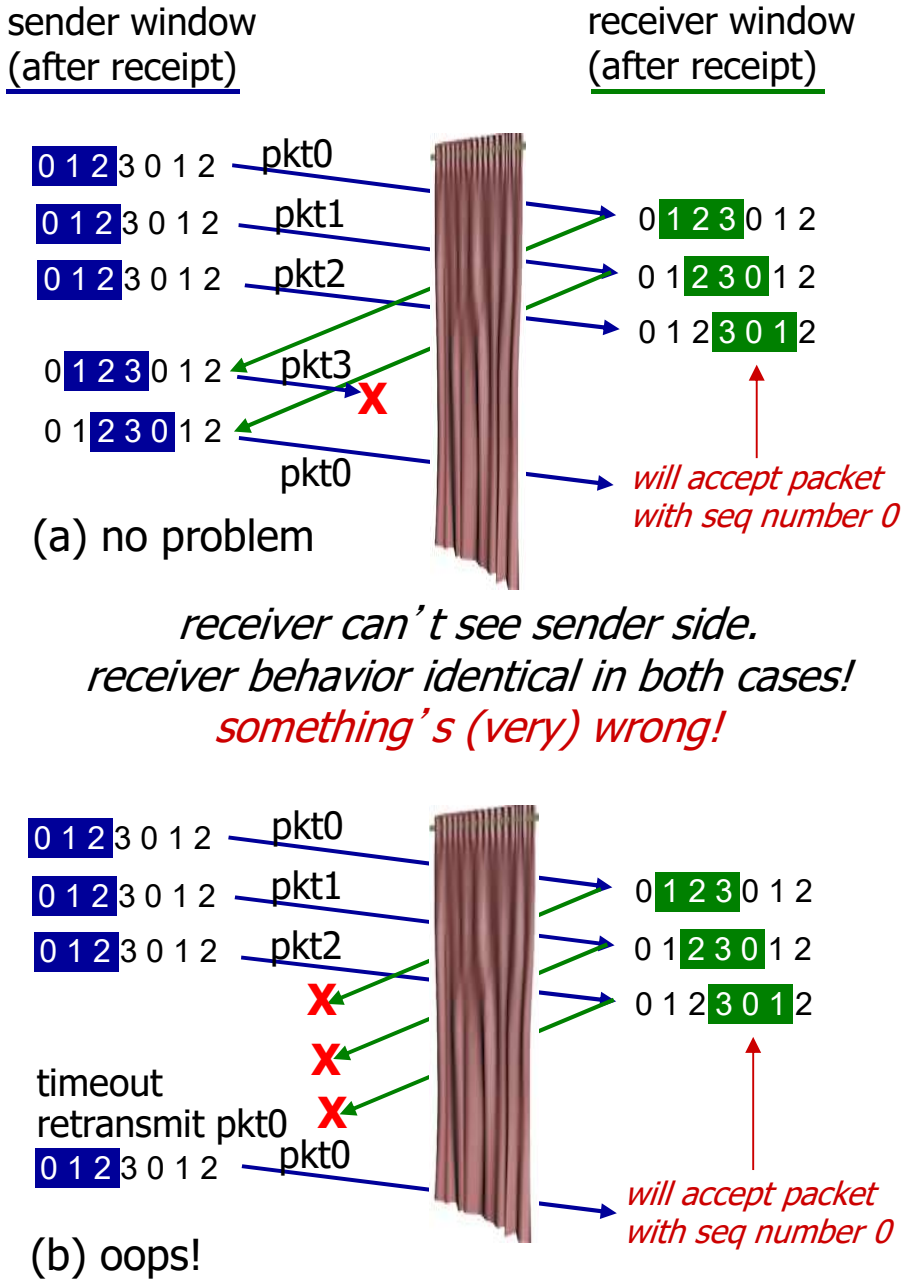


Selective repeat: dilemma

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)?



After-study Test :

- 순서번호가 가질 수 있을 값의 범위는 [0, 255]이다.
  ① 순서번호 필드의 크기는 몇 비트인가?
  ② 선택적 반복에서 최대 송신 윈도우 크기는 얼마인가?
  ③ GBN의 최대 송신 윈도우 크기는 얼마인가?