Trao đổi trực tuyến tại:

http://www.mientayvn.com/chat_box_li.html

CƠ SỞ KỸ THUẬT TRUYỀN SỐ LIỆU

Chương 4: Cơ sở của giao Thức

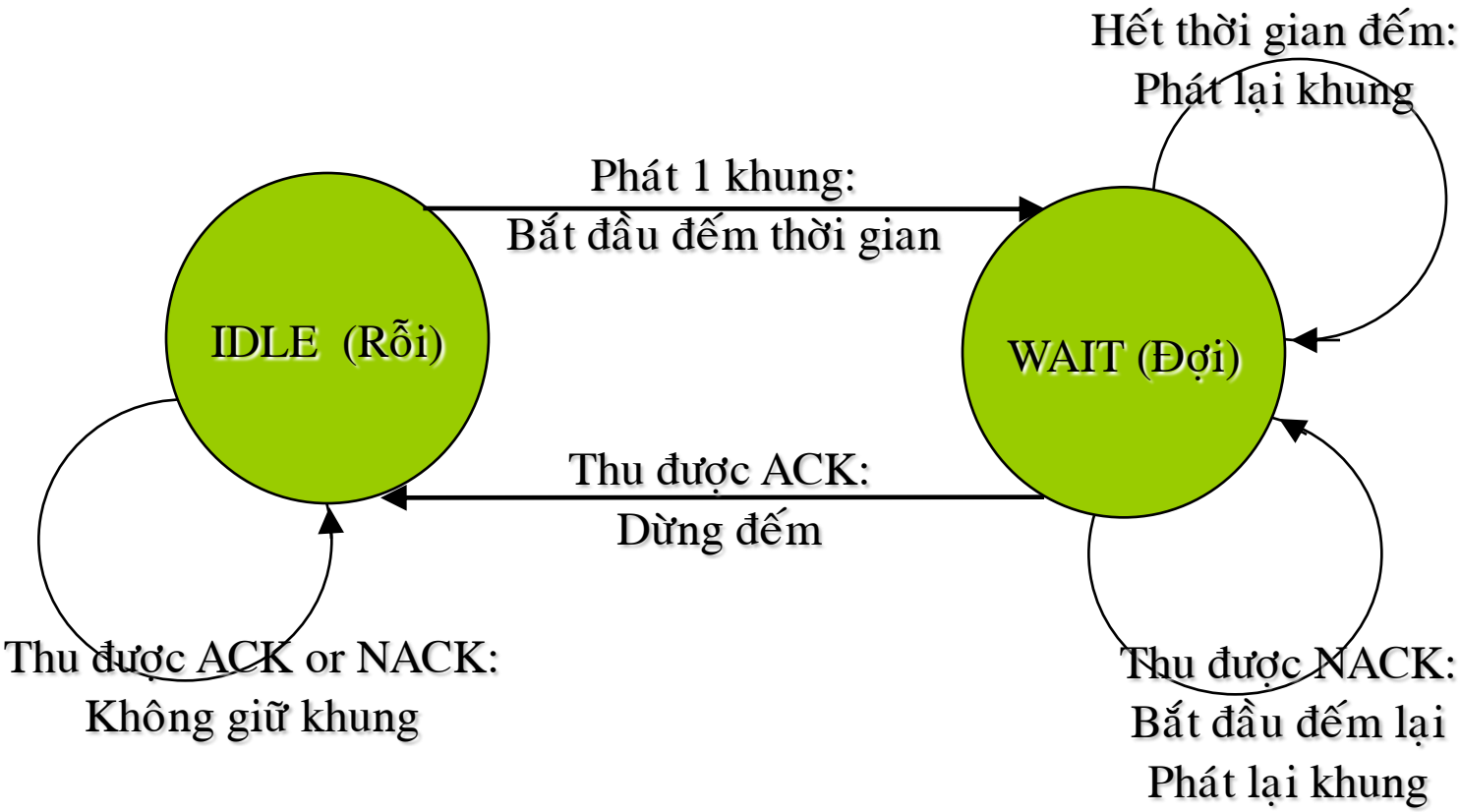
Nội dung chính

- Các phương pháp điều khiển lỗi bằng phát lại
  1. Điều khiển lỗi bằng Idle RQ
  2. Điều khiển lỗi bằng Continuous RQ
     - Selective repeat
     - Go – back - N
- Các phương pháp điều khiển luồng.
  1. X-OFF / X-ON
  2. Cửa sổ trượt
- Hiệu suất của liên kết.

Stop and Wait ARQ

- Nguồn S_1, phát đi 1 khung và bắt đầu đếm thời gian
- Đích, S_2, thu 1 khung
- Đích báo phát bằng một ACK
- Nguồn thu được ACK và dừng đếm
- Bây giờ, nguồn sẵn sàng bắt đầu cho 1 chu kỳ mới để phát 1 khung mới
- Nếu bộ đếm thời gian kết thúc trước khi nguồn nhận được ACK từ đích thì việc phát khung cũ sẽ được lặp lại.

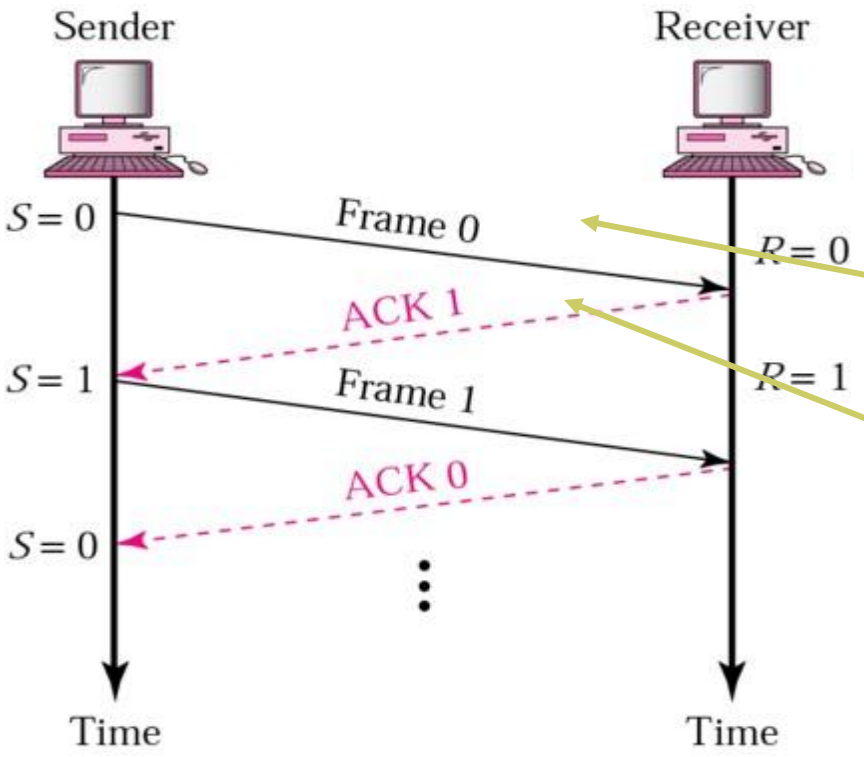
Stop and Wait ARQ



A Simplex Stop-and-Wait ARQ

1. normal operation
2. the frame is lost
3. the ACK is lost
4. the ACK is delayed

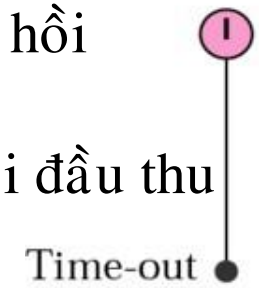
Stop-and-Wait ARQ
-Normal operation-



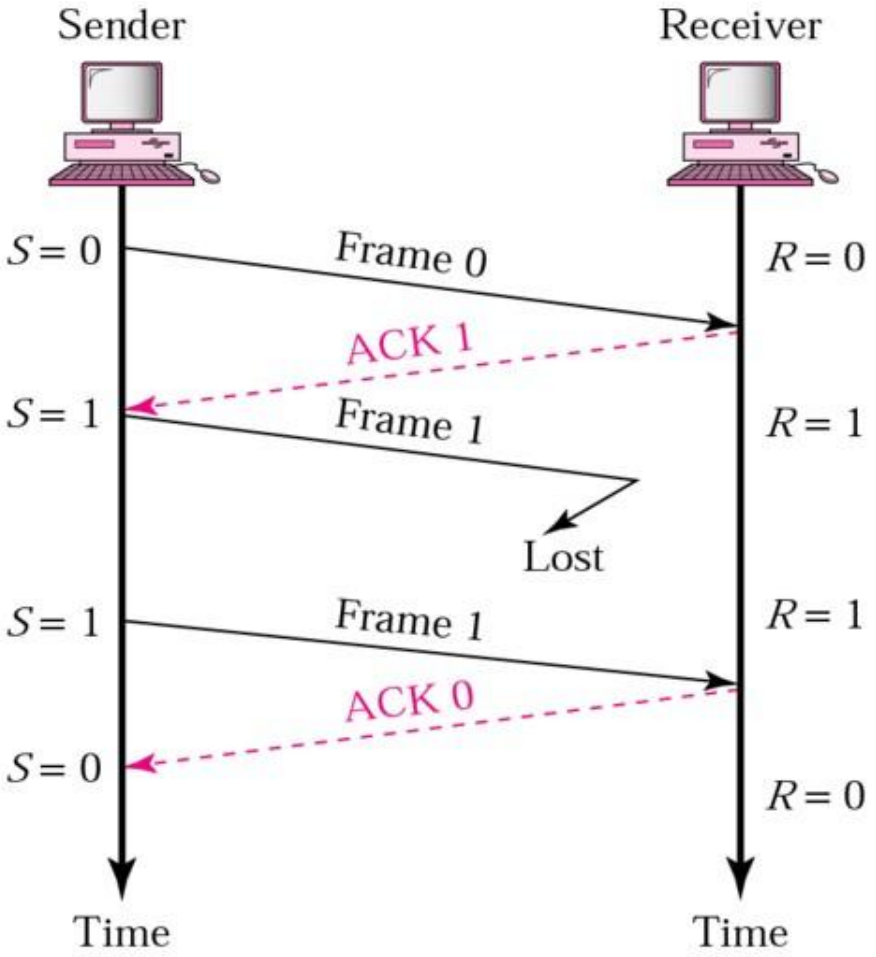
- Sender sẽ không gửi khung tiếp theo nếu không chắc chắn khung trước đó được nhận đúng.
- Số tuần tự cần thiết để kiểm tra khung nhận được là mới hay cũ.
- ACK – khi khung đúng và NACK - khi khung hỏng.

Stop-and-Wait ARQ – Lost or damaged frame-

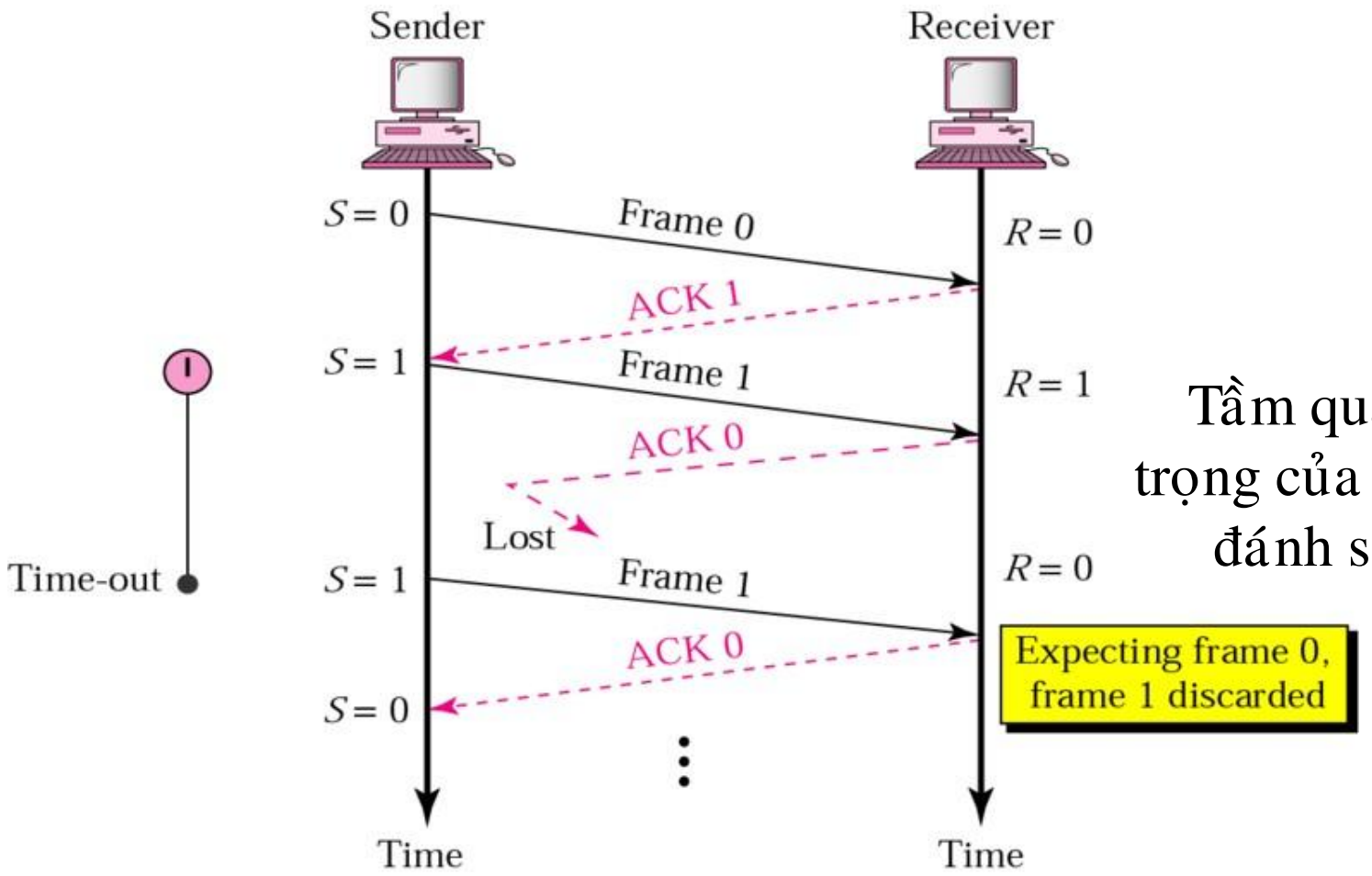
Trễ khứ hồi
+
Xử lý tại đầu thu



Should be as short as possible. Why?

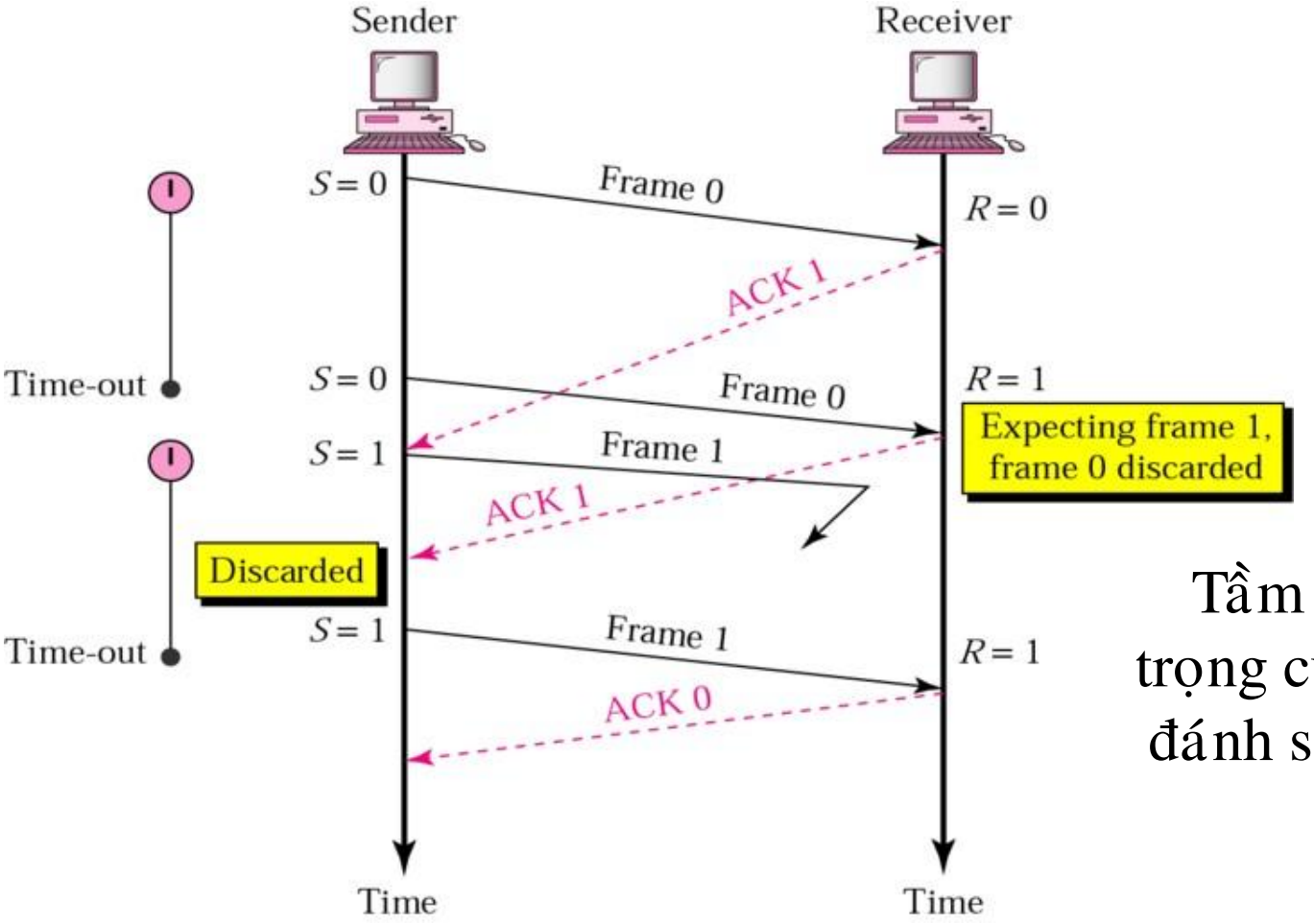


Stop-and-Wait ARQ - Lost ACK-



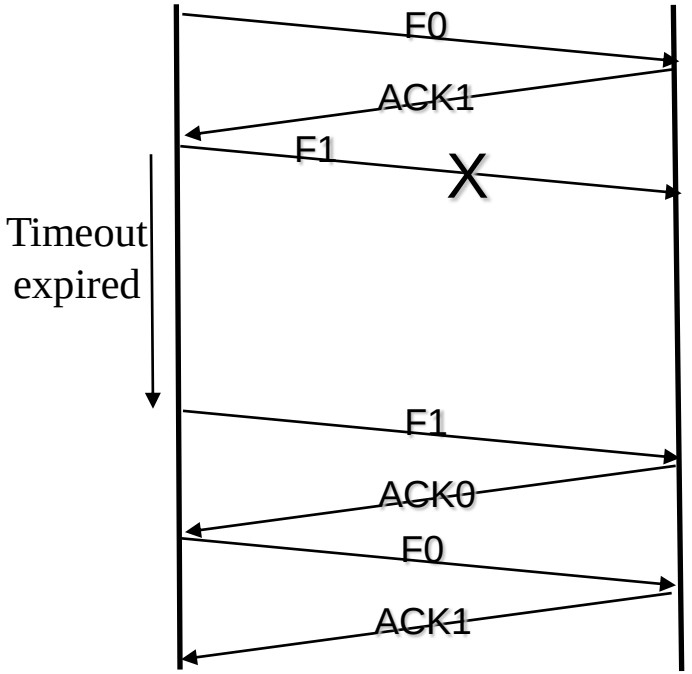
Tầm quan trọng của việc đánh số

Stop-and-Wait ARQ -Delayed ACK-

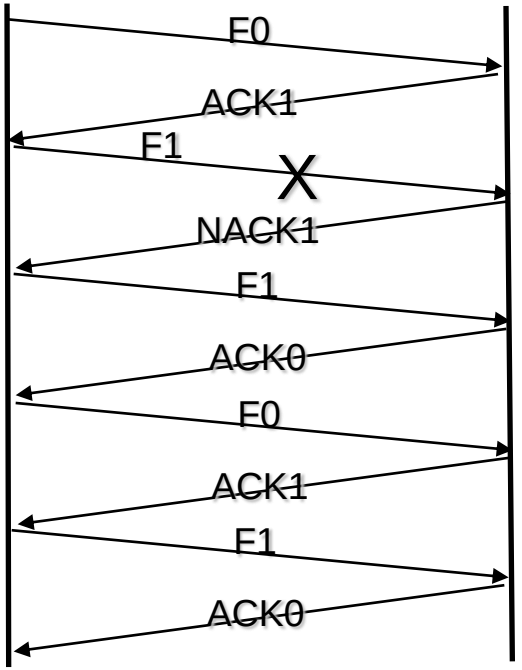


Tầm quan trọng của việc đánh số ACK

Stop and Wait ARQ: (3)



Stop and wait without NACK

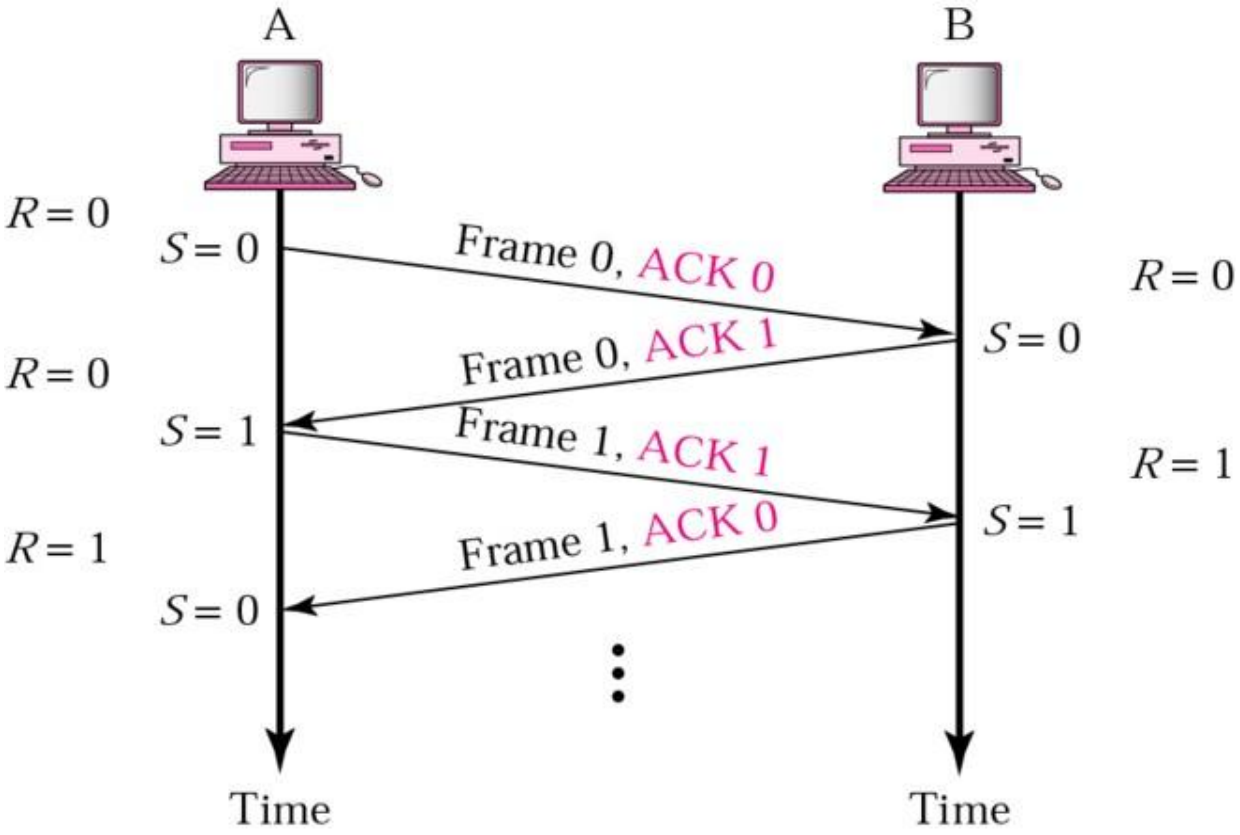


Stop and wait with NACK

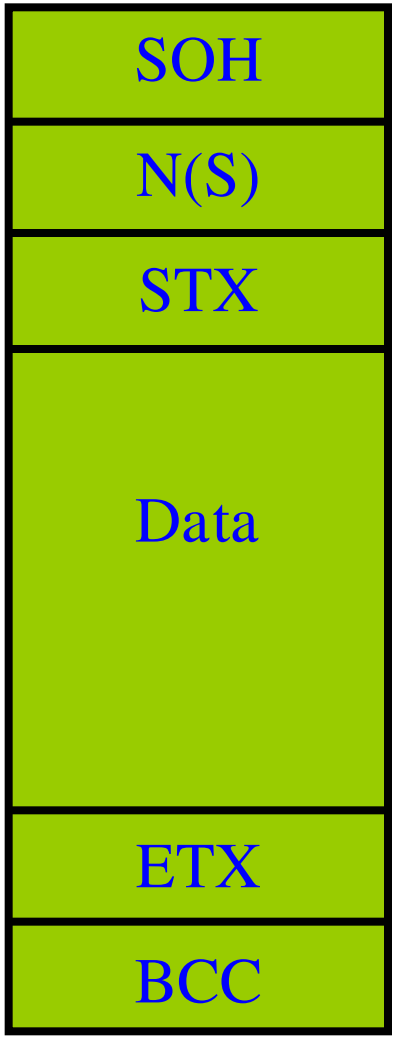
- Lỗi khung (Data bị sai)
  - Sử dụng NACK để cải tiến hiệu suất. NACK được thu trước khi hết timeout.

Duplex Stop-and-Wait ARQ

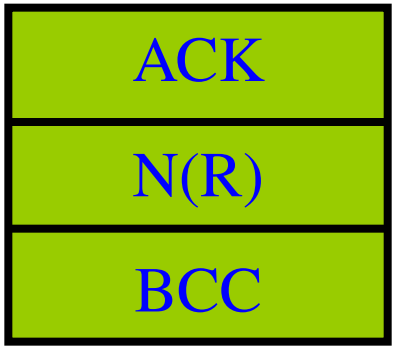
- Kết hợp
  - Kết hợp data with ACK (giảm overhead & tiết kiệm BW)



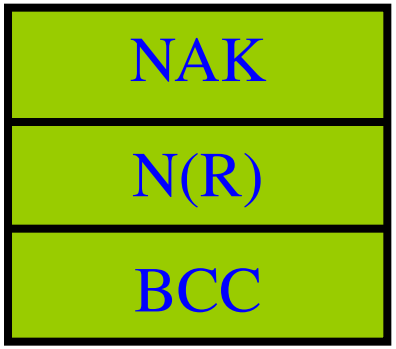
Cấu trúc các loại khung



I - Frame



ACK - Frame



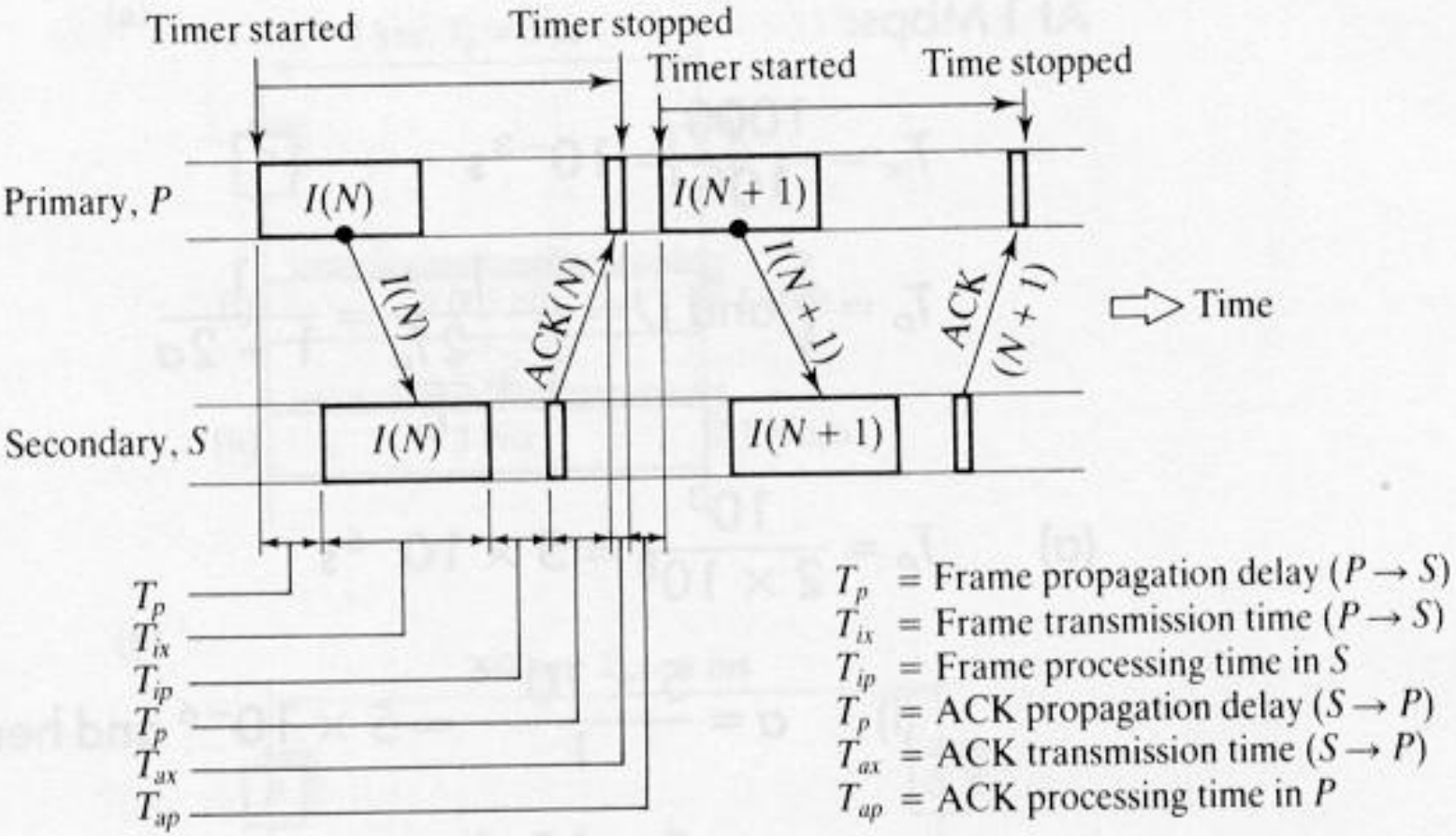
NAK - Frame

Hiệu suất Data Link Protocol

- Hiệu suất của giao thức:

$$U = \frac{\text{Thời gian phát 1 I-Frame, } T_{ix}}{\text{Tổng thời gian dùng cho phát 1 I-Frame, } T_t}$$

- Giả thiết:
  - Thời gian xử lý I-Frame tại S là không đáng kể.
  - Thời gian phát tại S and xử lý tại P các ACK\NAK Frame là không đáng kể.



Các loại thời gian được sử dụng để tính U

Hiệu suất của Idle RQ

- T_{ix}: Thời gian để phát 1 I-Frame
- T_p: Thời gian lan truyền qua kênh
- N_f: Số lần trung bình phải phát để thu được mà không có lỗi.

$$Efficiency(\eta) = \frac{T_{ix}}{N_f\,(T_{ix} + 2T_p)}$$

Hiệu suất của Stop and Wait ARQ

- Gọi P_B là xác suất để 1 bit có lỗi (BER).
  - Xác xuất để 1 bit không lỗi là $(1 - P_B)$
  - Xác xuất để 1 khung độ dài N_i không lỗi là $(1 - P_B)^{Ni}$.
  - Xác xuất để 1 khung độ dài N_i có lỗi là $P_f = 1 - (1 - P_B)^{Ni}$.
  - Xác xuất để 1 khung không lỗi là: $1 - P_f$.

Hiệu suất của Stop and Wait ARQ

- Giả thiết ACK or NACK không mất hoặc hỏng, và timeout không sảy ra, khi đó:
  - Số lần phát trung bình một khung đề không có lỗi là

$$N_f = 1 / (1 - P_f)$$

  - Vậy hiệu suất của giao thức là:

$$U = \frac{(1 - P_f) . T_{ix}}{T_{ix} + 2 T_p}$$

Hạn chế của Stop-and-Wait ARQ

- Sau mỗi một khung gửi đi, Host phải chờ 1 ACK
  - Không hiệu quả sử dụng bandwidth

- Để cải thiện hiệu quả, ACK nên được gửi sau một số khung, gọi là Continuous ARQ.

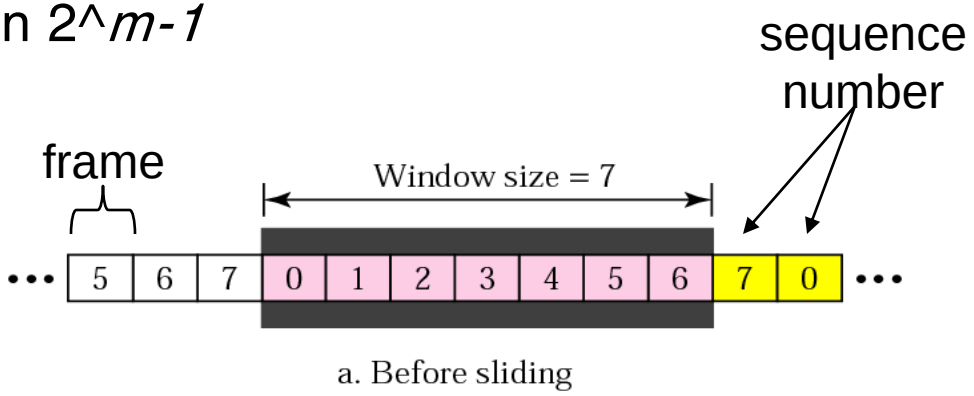
- 2 loại Continuous ARQ (Sliding Window protocols):
  1. Go-back-*N* ARQ
  2. Selective Repeat ARQ

.

Sliding Window Protocols

- Số tuần tự
  - Các khung phát được đánh số tuần tự
  - Số tuần tự được lưu ở Header của khung
    - Nếu số bít đánh số tuần tự ơ' header là *m* thì số tuần tự đếm từ 0 đến 2^*m-1*

- Cửa sổ trượt
  - Để lưu các khung chưa báo phát
  - Kích thước của sổ đầu thu có thể bằng hoặc lớn hơn 1.



a. Before sliding

Caâu hoûi

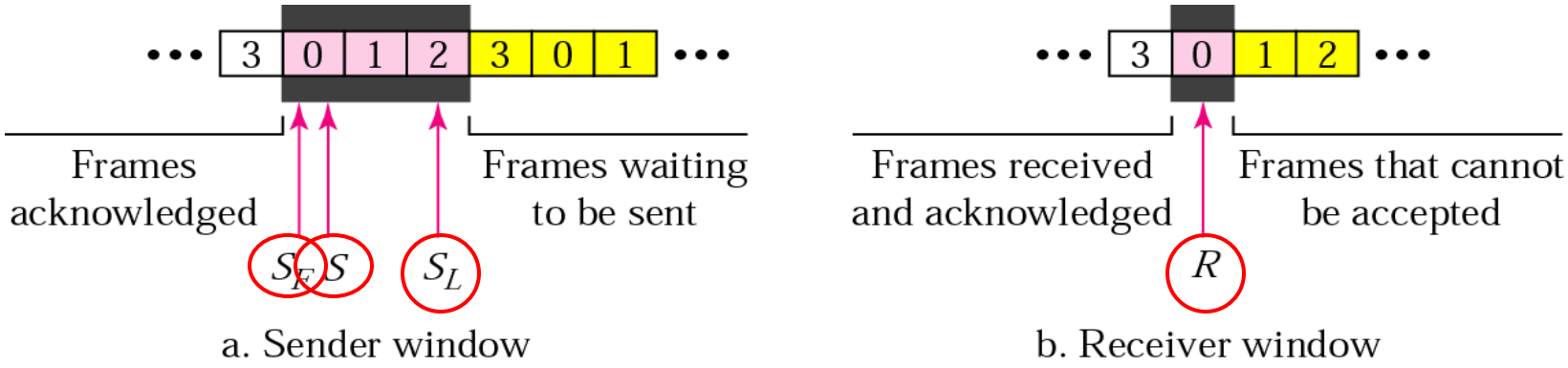
- Kích thước của sổ (WS) của máy thu có ảnh hưởng như thế nào đến thứ tự của các gói thu được?

Traû lôøi:

- WS = 1 có nghĩa là các gói phải thu đúng theo thứ tự!
- Điều này không đúng với WS > 1

Go-back-*N*- Control variables-

- *S*- chỉ số tuần tự của khung đang được phát
- S_F – chỉ số tuần tự của khung đầu tiên trong cửa sổ
- S_L – chỉ số tuần tự của khung cuối cùng trong cửa sổ
- *R* – chỉ số tuần tự của khung đang chờ thu



The name of Go-back-*N*: why?

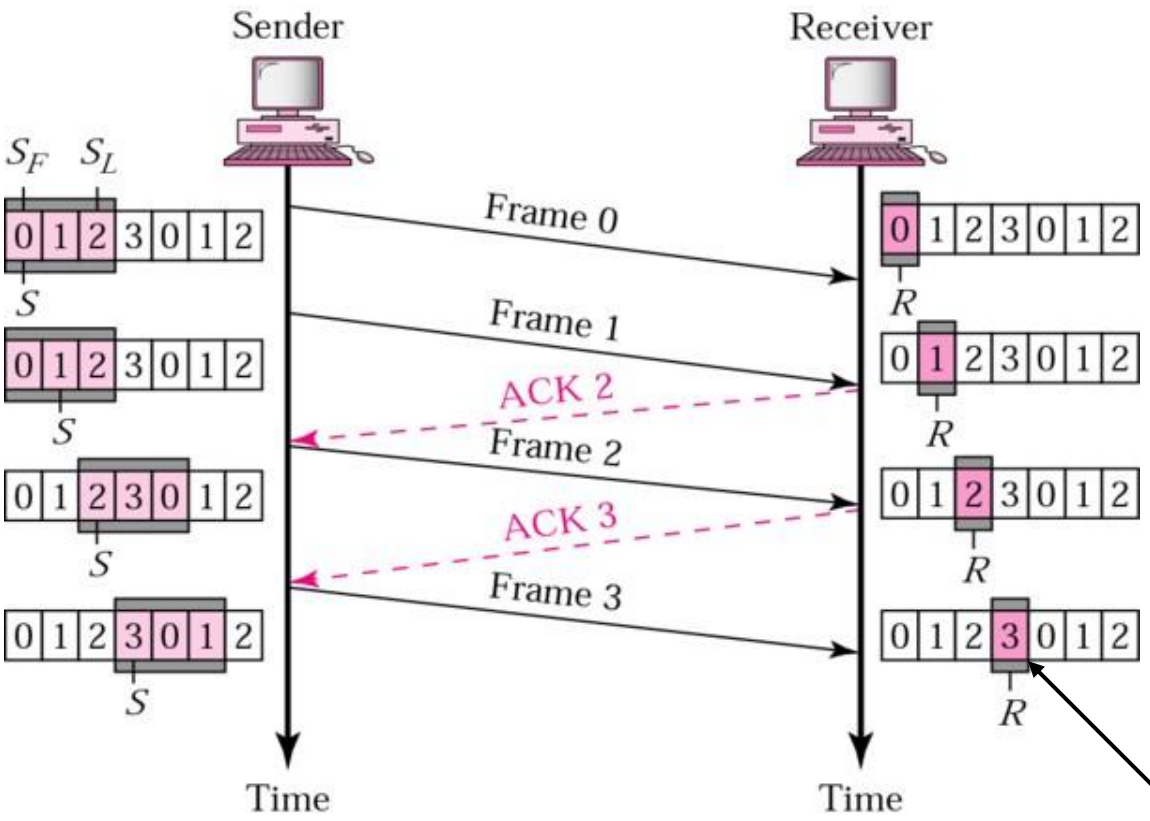
- Phát lại khung
  - Khi một khung bị hỏng, đầu phát sẽ **quay lại** và phát lại một tập hợp các khung tính từ khung không có báo phát (ACK)
  - Số lượng khung đượcphát lại là ***N***

Example:

WS là 4.

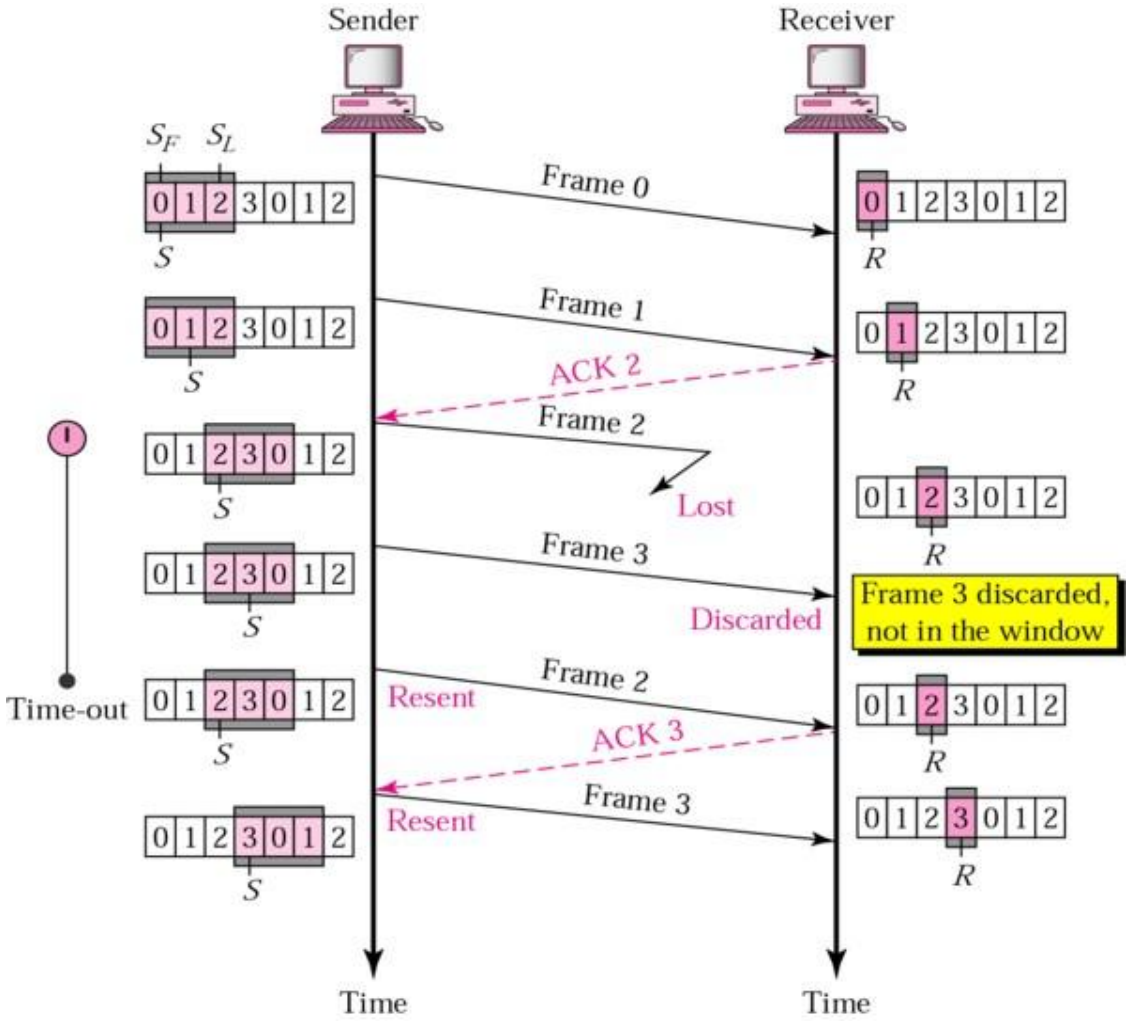
Đầu phát vừa phát khung 6 và hết thời gian đếm khung 3 (khung 3 không có ACK). Đầu phát sẽ phát lại các khung 3, 4, 5, 6.

Go-back-*N*- normal operation-



- Có bao nhiêu khung co 1 thể phát mà không chờ ACK?
- ACK1 – không cần thiết nếu ACK2 được phát đi.

Go-back-*N* - damaged or lost frame-

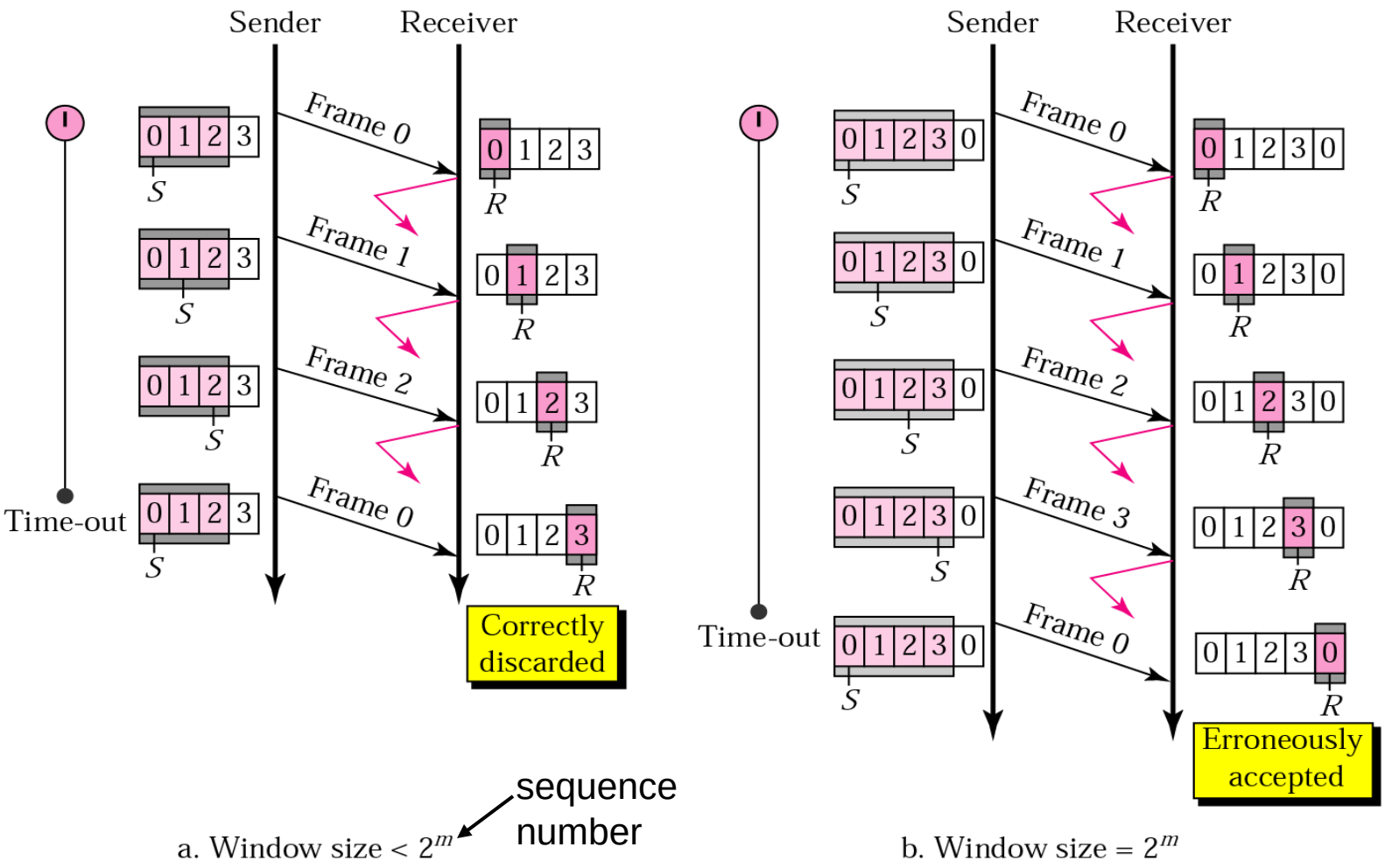


Các khung hỏng bị loại bỏ!

Tại sao các khung vẫn thu đúng thứ tự tuy không có bộ đệm?

Nhược điểm của phương pháp này là gì?

Go-back-*N*- sender window size-



Go-back-*N*

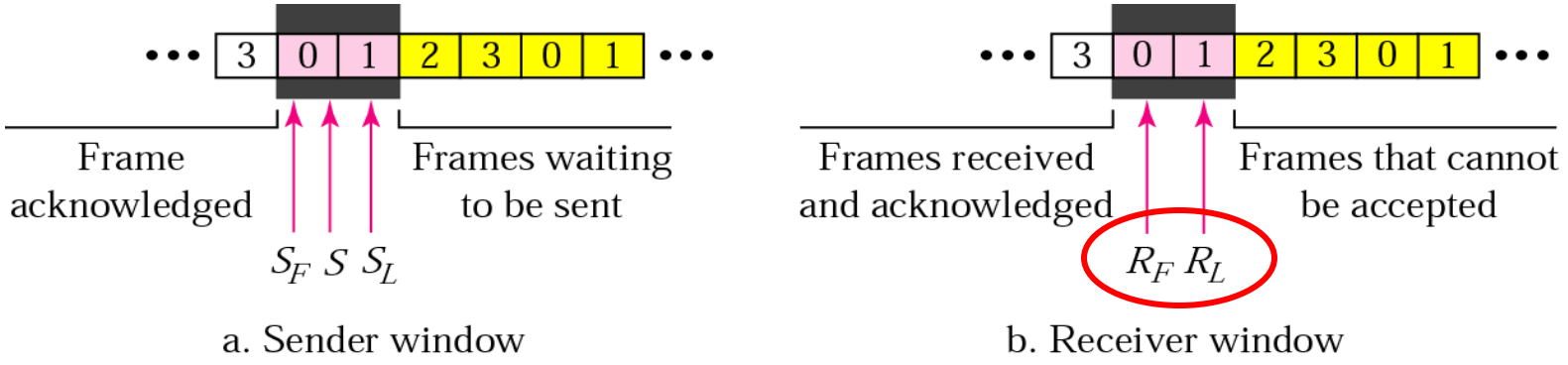
- Không hiệu quả
  - Tất cả các khung không đúng thứ tự đều phải phát lại
- Nếu liên kết có tạp âm sẽ gây ra vấn đề:
  - Nhiều khung phải phát lại -> tốn bandwidth

- Giải pháp
  - Chỉ phát lại những khung hỏng

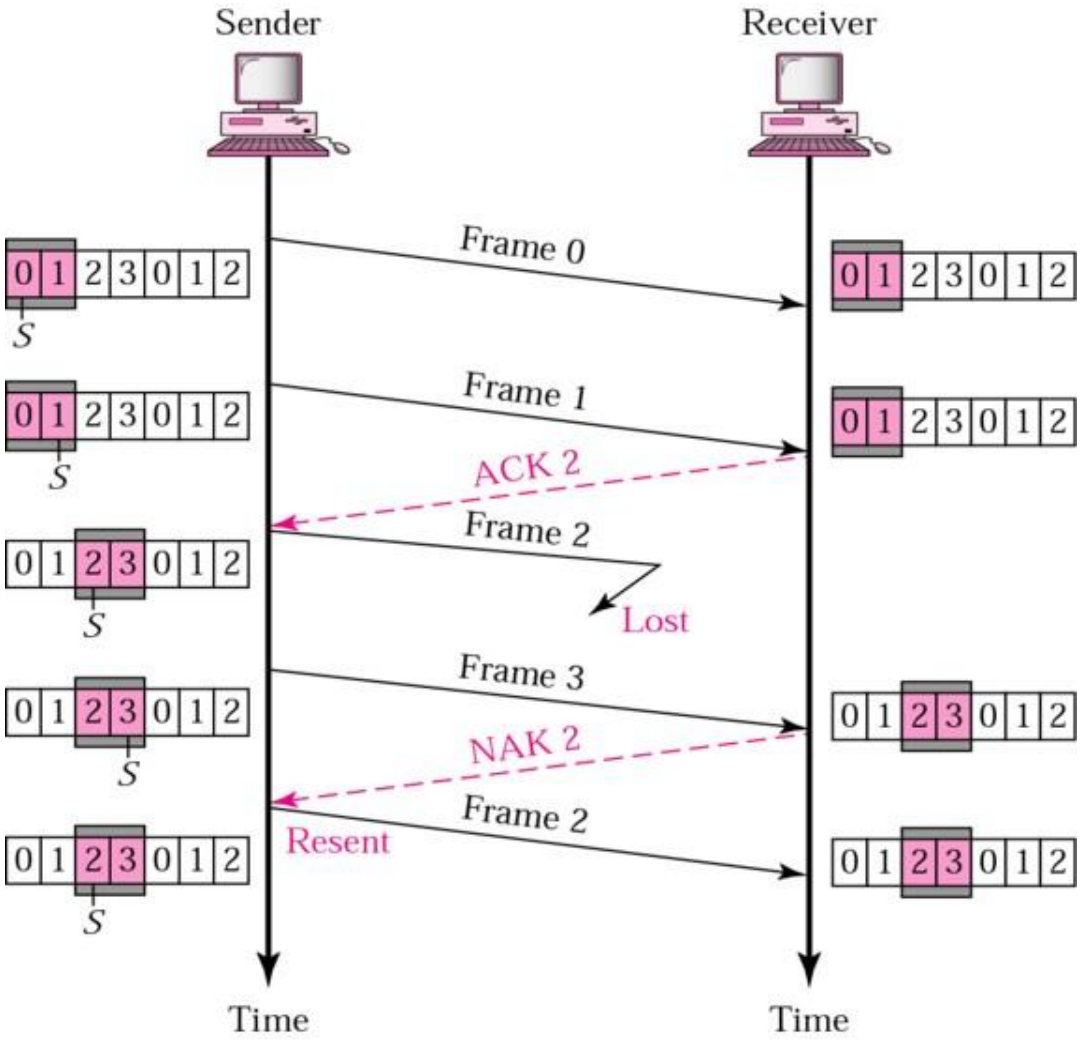
- Selective Repeat ARQ
  - Chống lại việc phát lại không cần thiết

Selective Repeat ARQ

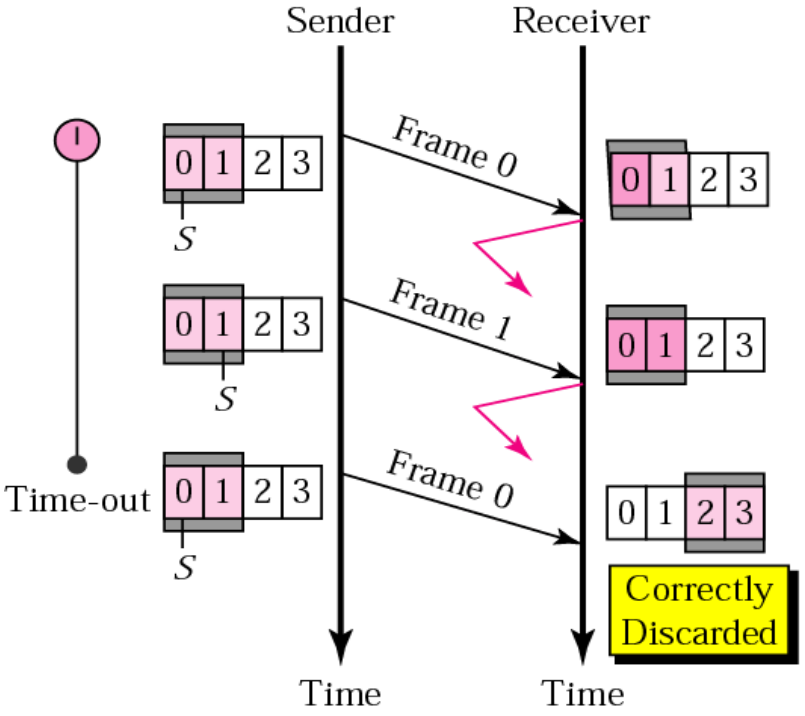
- Xử lý tại đầu thu phức tạp hơn
- Kích thước cửa sổ giảm xuống <= 2^{m-1}
- Đầu phát và đầu thu có kích thước của sổ như nhau
- Đầu thu chờ thu một tập hợp khung trong một phạm vi của số tuần tự



Selective Repeat ARQ - lost frame-

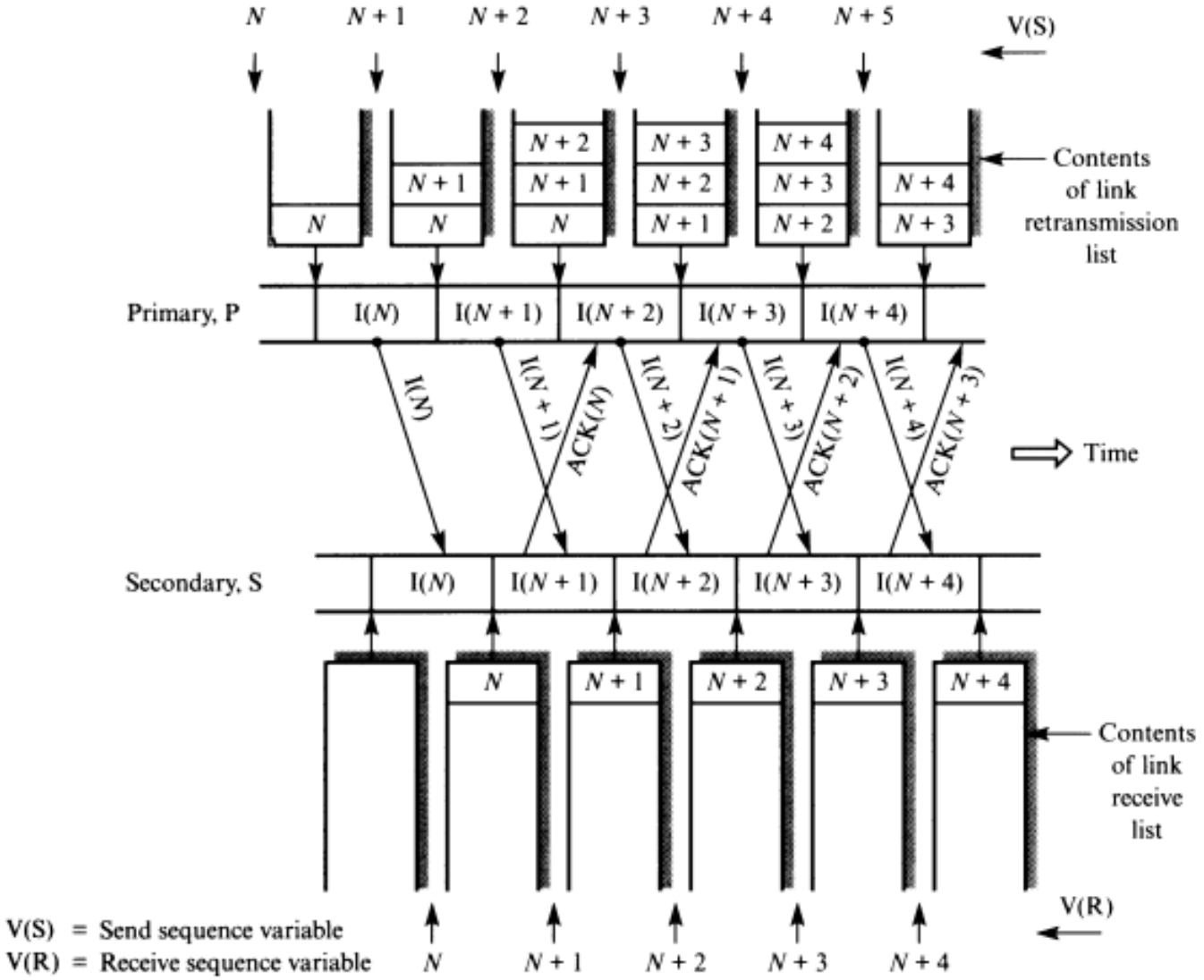


Selective Repeat ARQ-sender window size-



a. Window size = 2^{m-1}

Basic operation of Continuous RQ protocol

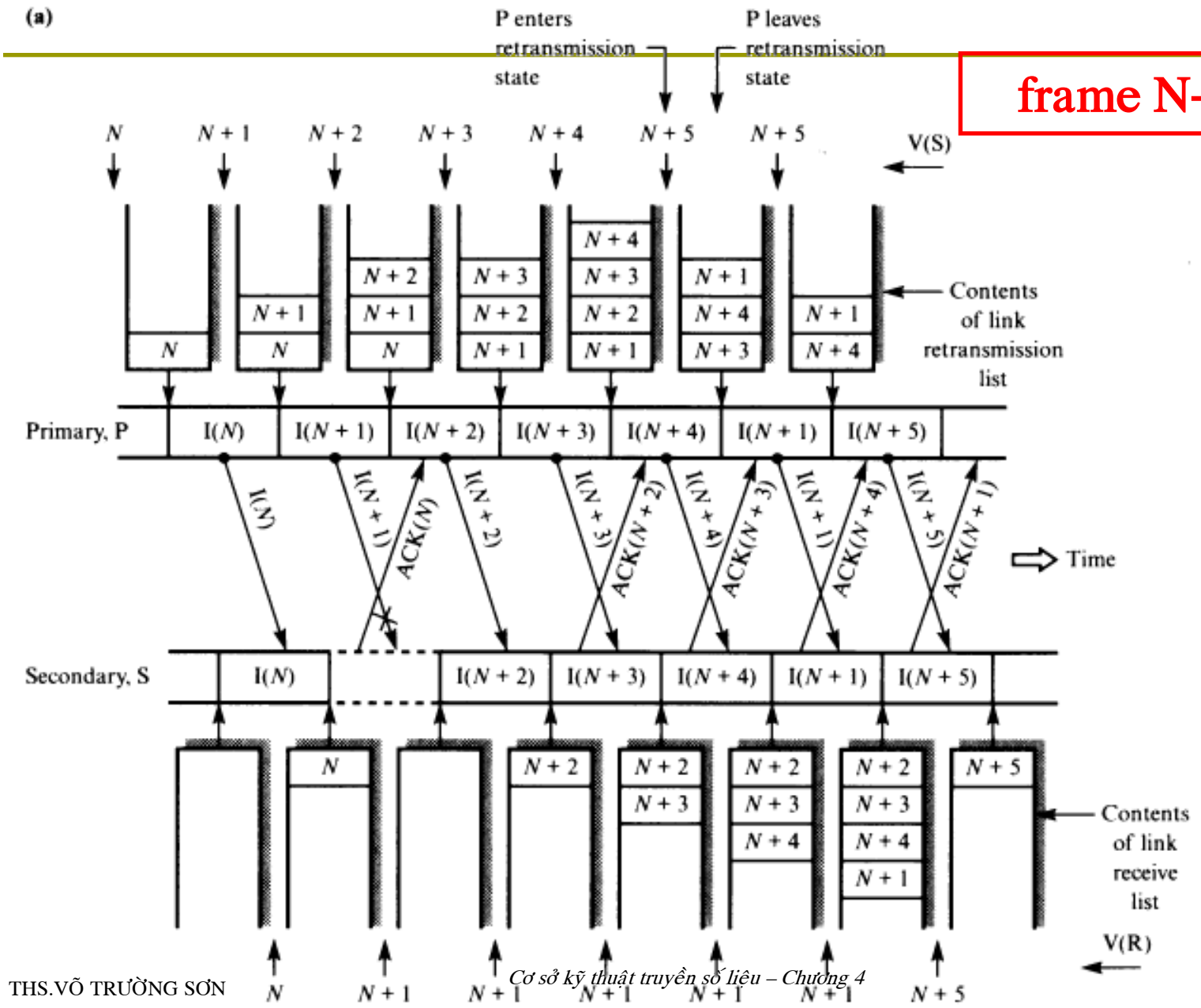


Operation of Continuous RQ protocol with Selective Repeat

(a) Corrupted I-frame

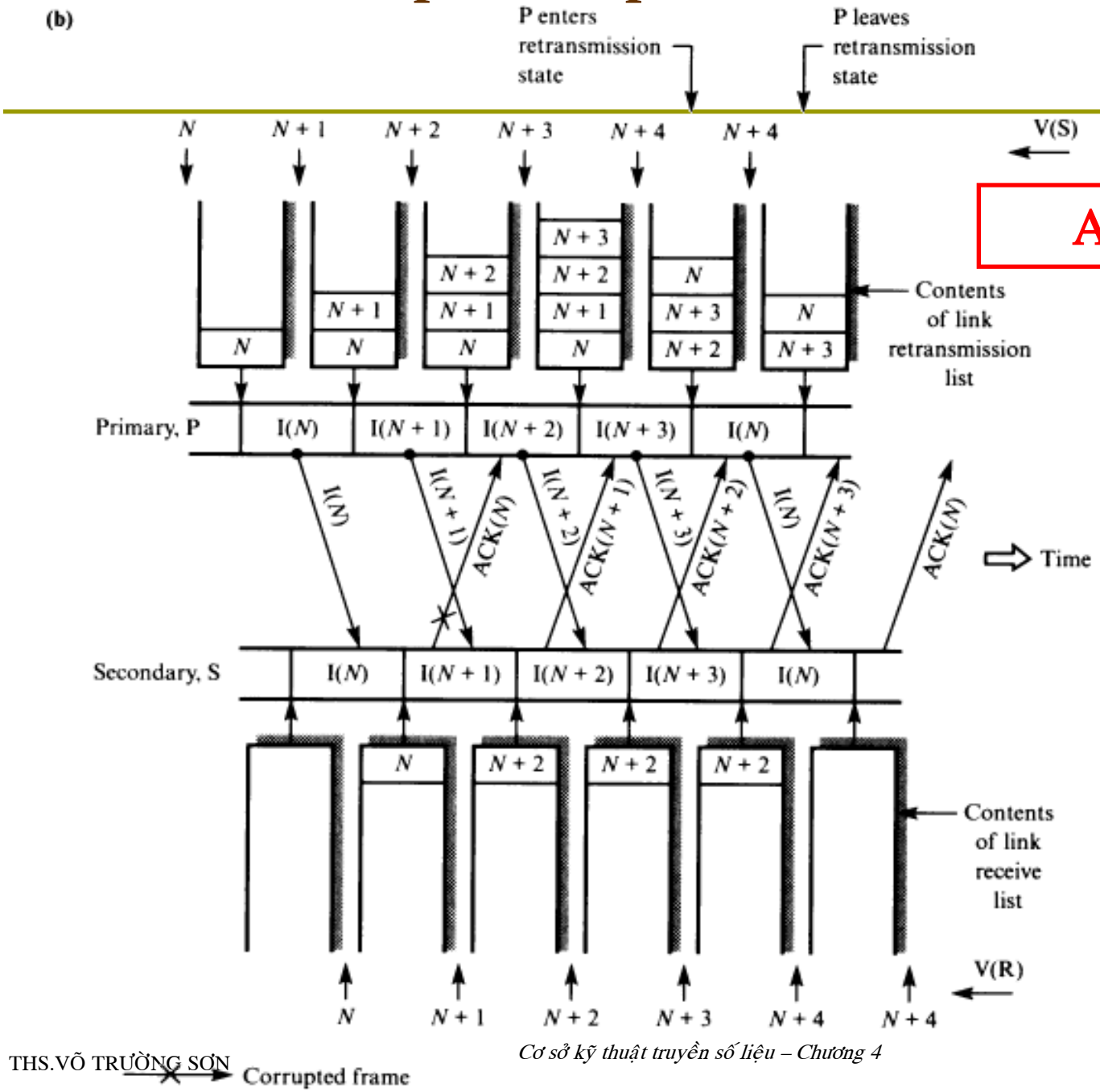
(b) Corrupted ACK frame

Selective Repeat [implicit retransmission]



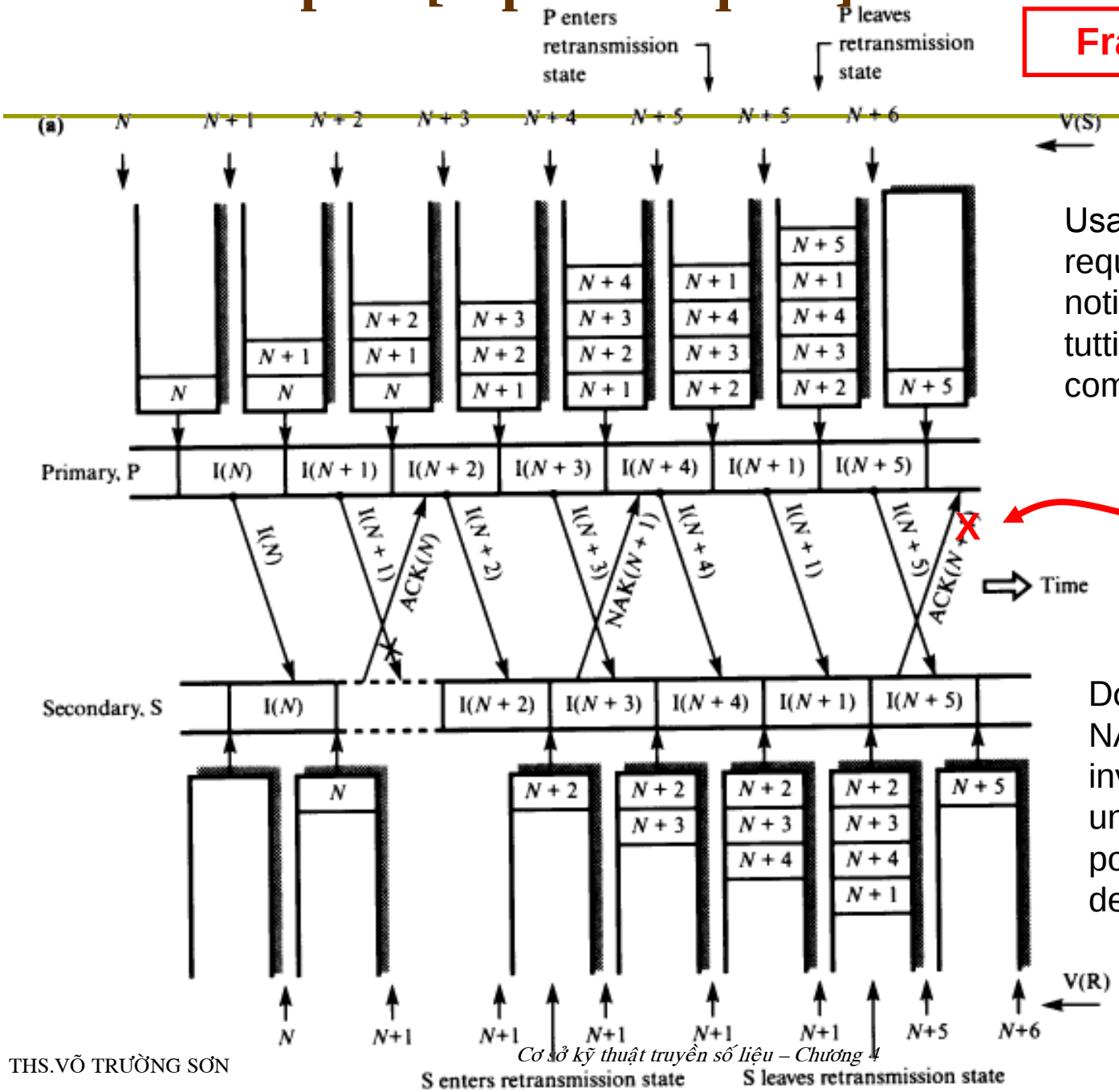
frame N+1 hỏng

Selective Repeat [implicit retransmission]



ACK-N hỏng

Selective Repeat [explicit request]



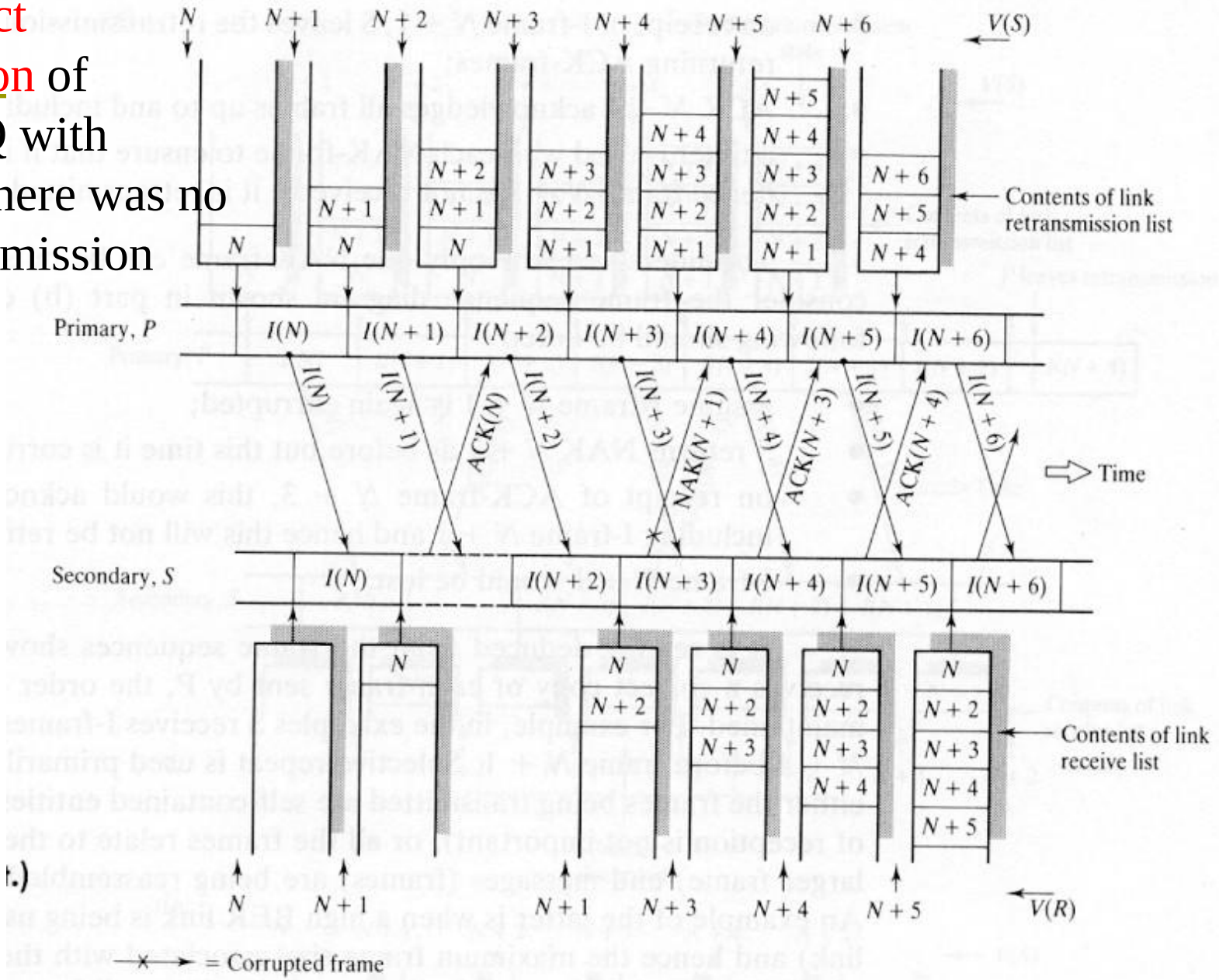
Frame N+1 Hỏng

Usando l'explicit request, ACK(N) notifica la ricezione di tutti i frame fino ad N compreso!

ACK(N+4)

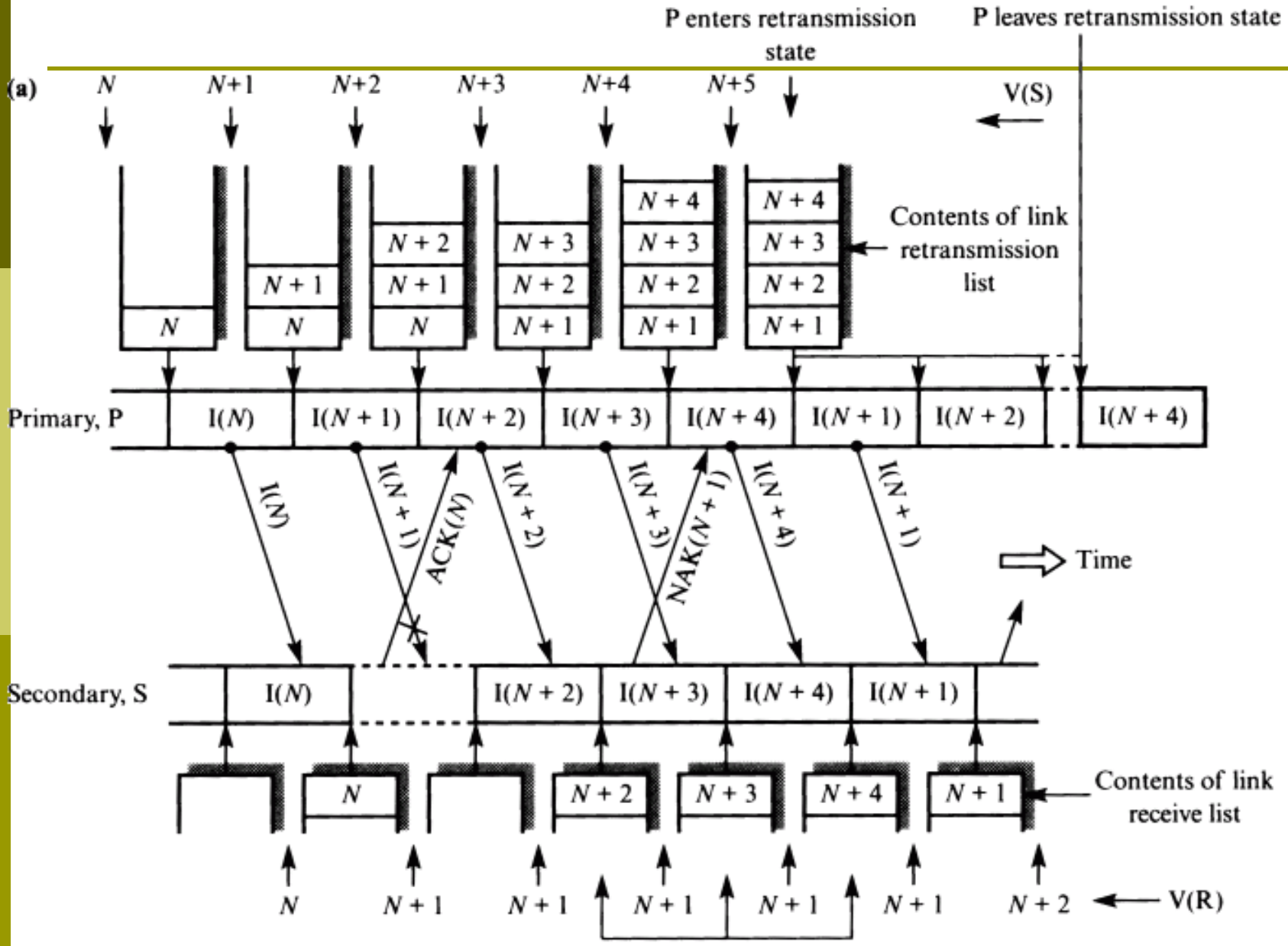
Dopo l'invio di un NAK, S **smette** di inviare ACK. Altrimenti un NAK corrotto porterebbe alla perdita del pacchetto!

Incorrect operation of Cts. RQ with S\R if there was no Retransmission State



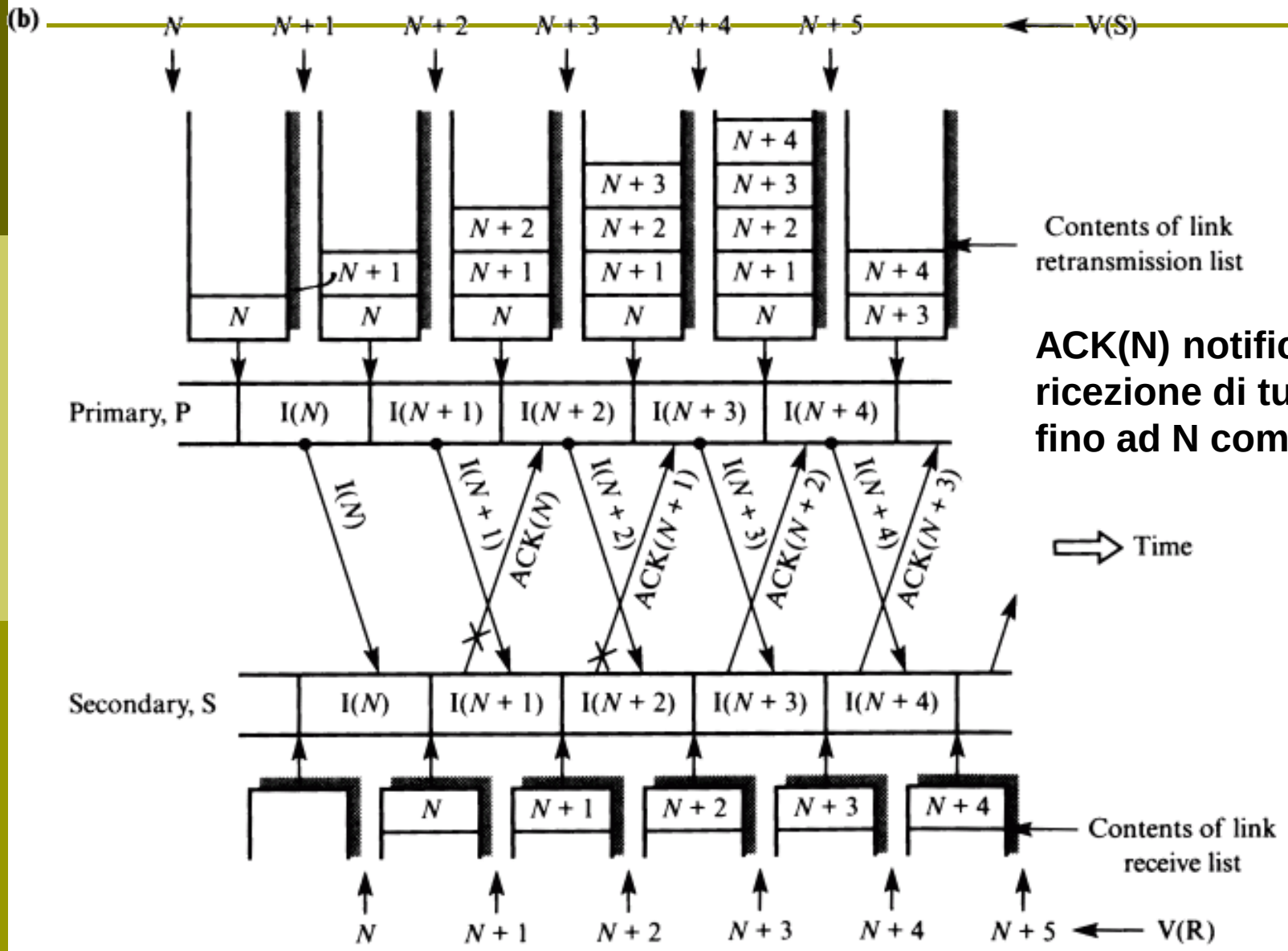
Go Back N

Frame N+1 Hong



Go Back N

ACK N & N+1 Hong



ACK(N) notifica la ricezione di tutti i frame fino ad N compreso!

Comparison of Protocol Buffer Requirements

	P Buffer Size	S Buffer Size
Idle RQ	1	1
Cts. RQ - SR	K	K
Cts. RQ - Go Back N	K	1

Cơ số đếm = P buffer size + S buffer size

Go Back N Utilization: 1

- If errors and losses make retransmissions necessary then

$$U = T_f / (N_{rsw} \times T_t)$$

If no timers expire, $N_{rsw,}$, *is* the average number of transmissions per frame, N_a. Otherwise N_{rsw} is a function $f(N_a)$

- Let P_f be the probability that a given frame contains errors.
  - The probability that it will take i attempts to transmit the frame is

$$P_f^{i-1}(1-P_f)$$

  - Each error requires that K frames be retransmitted
  - The average number of retransmissions N_a

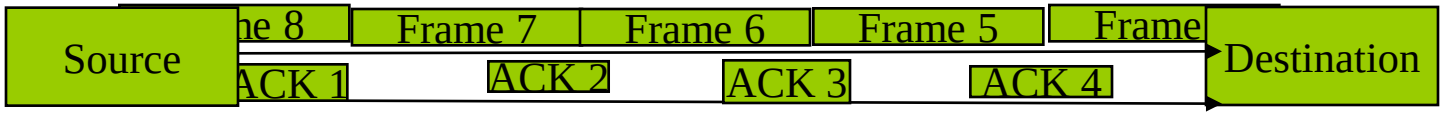
(original transmission and *i*-1 retransmissions of K frames)

$$N_a = \sum_{i=1}^{\infty}[1+(i-1)K]\ P_f^{i-1}(1-P_f) = \frac{1-P_f+KP_f}{1-P_f}$$

What is K?

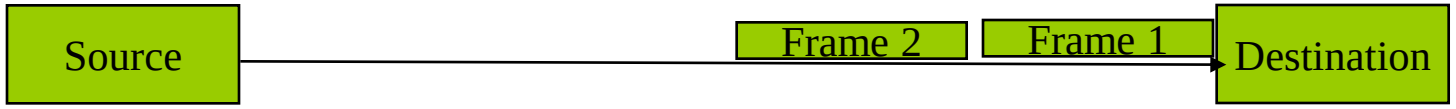
- Round trip travel time means T_t / T_{FRAME} frames will be transmitted before the ACK for the first frame is received.
- For a window length $W \geq T_t / T_{FRAME}$

  $\therefore K \approx T_t / T_{FRAME}$ $(=2a+1)$



W = 8 a>4 K≈9

- For a window length $W < T_t / T_{FRAME}$ $\therefore K \approx W$



W = 2 K≈2

Go Back N ARQ Utilization: 2

$$U = T_f / (N_r T_t) = \frac{(1 - P_f) T_f}{(1 - P_f + K P_f) T_t}$$

- Now substitute our estimates of K

$$K = \begin{cases} W & W < T_t / T_{FRAME} \\ T_t / T_{FRAME} & W \geq T_t / T_{FRAME} \end{cases}$$

- To give

$$U = \begin{cases} \dfrac{(1 - P_f) T_f}{(1 - P_f + W P_f) T_t} & W < T_t / T_{FRAME} \\ \dfrac{(1 - P_f)}{(1 - P_f + \frac{T_t}{T_{FRAME}} P_f)} & W \geq T_t / T_{FRAME} \end{cases}$$

Go Back N ARQ Utilization: 3

- Assume that
  - Processing time is negligible, transmission time for Frames is » than for ACKs
  - no ACKs or NACKs are lost or damaged, and that the timeout timer never expires
  - Then

$$T_t / T_{FRAME} = (1 + 2a)$$

$$U = \begin{cases} \dfrac{W(1-P_f)}{(1-P_f+WP_f)(2a+1)} & W < 2a+1 \\ \dfrac{(1-P_f)}{(1+2aP_f)} & W \geq 2a+1 \end{cases}$$

- If the assumption is relaxed then the longer time for retransmission (timeout duration) must be included in the analysis. A separate probability for retransmission frames due to lost ACKs and NACKs may also be introduced

Selective Reject Utilization: 1

- If errors and losses make retransmissions necessary, then for sliding window flow control

$$U = T_f / (N_{rsw} \times T_t)$$

  If no timers expire, $N_{rsw,}$, *is* the average number of transmissions per frame, N_a. Otherwise N_{rsw} is a function $f(N_a)$

- Let P_f be the probability that a given frame contains errors.
  - The probability that it will take i attempts to transmit the frame is

$$P_f^{i-1}(1-P_f)$$

  - Each error requires that 1 frame be retransmitted
  - The average number of retransmissions N_a (original transmission and *i*-1 retransmissions)

$$N_a = \sum_{i=1}^{\infty} i\, P_f^{i-1}(1-P_f) = \frac{1}{1-P_f}$$

Selective Reject Utilization: 2

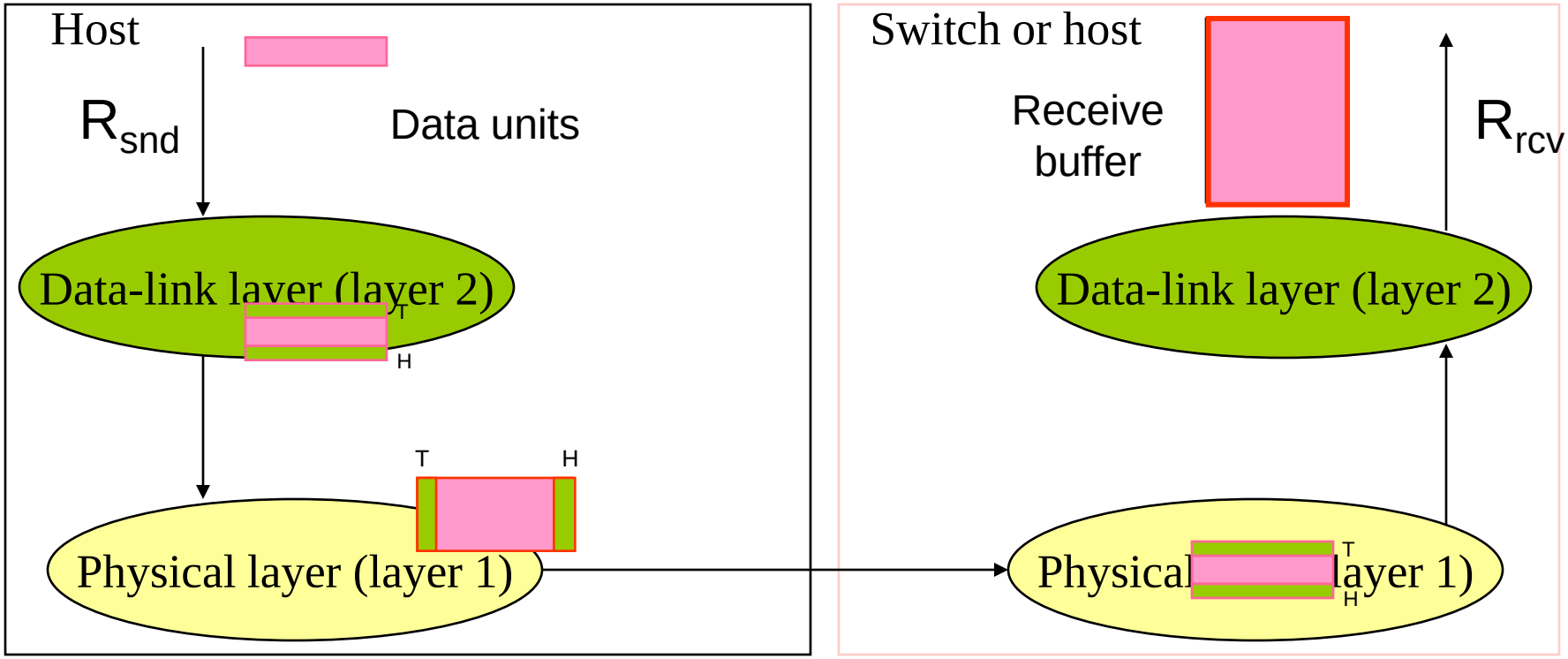
- Assume that no ACKs or NACKs are lost or damaged, and that the timeout timer never expires
  - Then N_r is the average number of transmissions per frame and the channel utilization is

$$U = (1-P_f)T_f / T_t$$

- Also assuming that
  - Processing time is negligible
  - The acknowledgment frame is small compared to the data frame

$$U = \begin{cases} W(1-P_f)/(1+2a) & W < (2a+1) \\ 1-P_f & W \geq (2a+1) \end{cases}$$

Flow control problem



- Rates of the transmitter and receiver at the physical layer are usually matched.
- The flow control problem arises because the higher layer (AL or NL) does not deplete out the buffer into which the DLL at the receiver writes at the same rate at which data is being passed to the DLL at the sender.

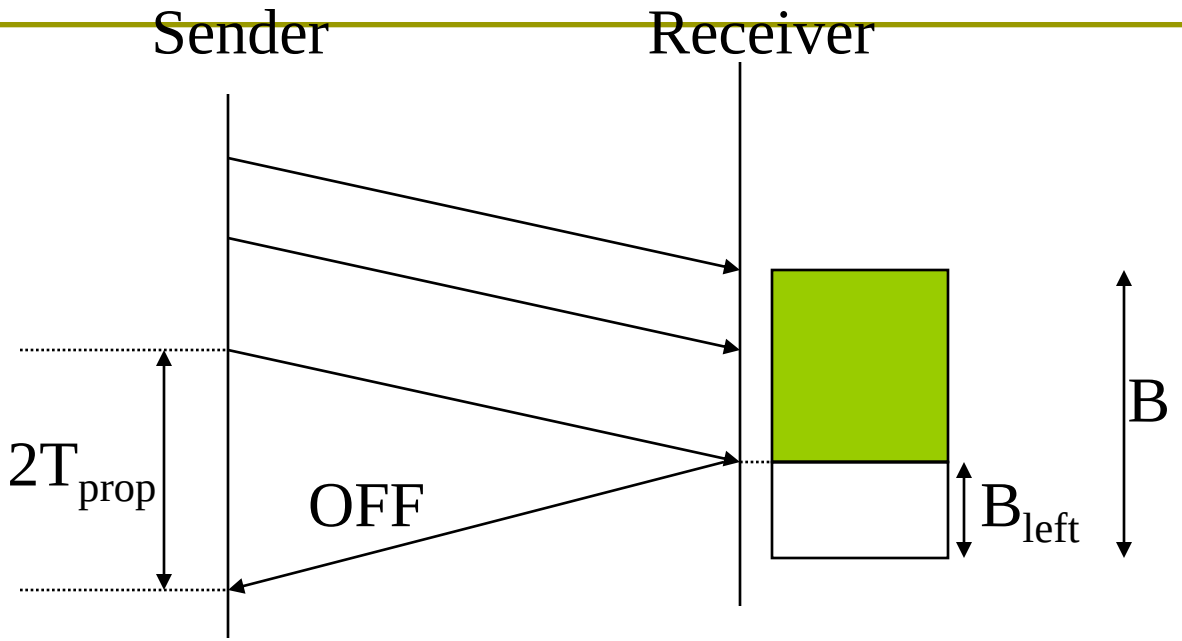
Techniques for flow control

- *Flow control* prevents buffer overflow by regulating rate at which source is allowed to send information
  - Stop-and-wait
  - ON-OFF
  - Window
  - Rate control

Stop-and-wait

- No problem if ACK is sent back after higher layer depletes the buffer holding the single frame's payload

ON /OFF



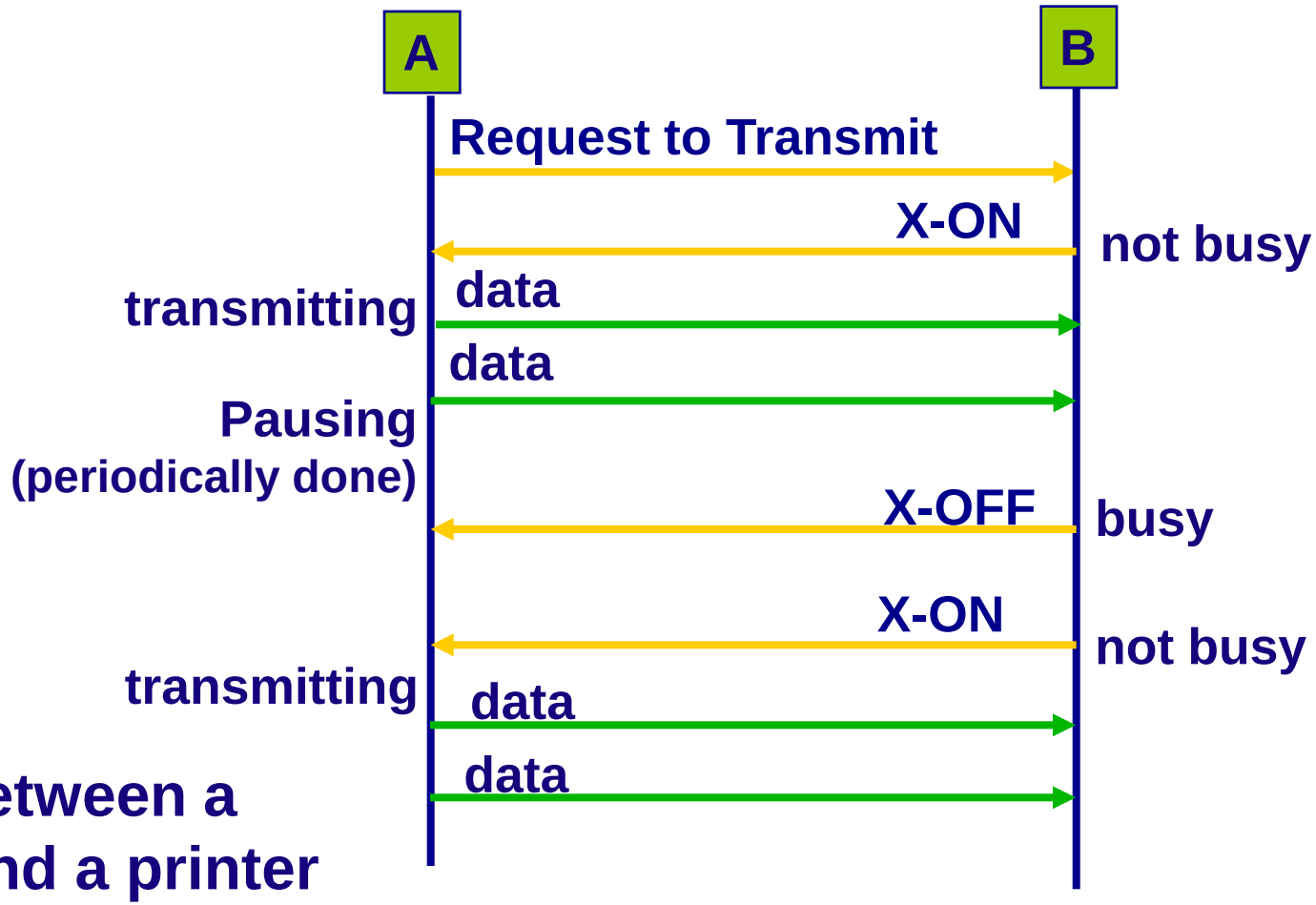
Link rate R_{snd}

$B_{left}=2T_{prop}(R_{snd} - R_{rcv})$

If R_{snd} is different from R_{rcv}, where R_{snd} is the rate at which the layer above the DLL at the sender sends data to that DLL and R_{rcv} is the rate at which the layer above DLL at the receiver depletes the DLL receive buffer, then the receiver should send the OFF signal when $B_{left} = 2T_{prop}(R_{snd}-R_{rcv})$

Điều khiển luồng bằng X-ON / X-OFF

An older controlled access protocol

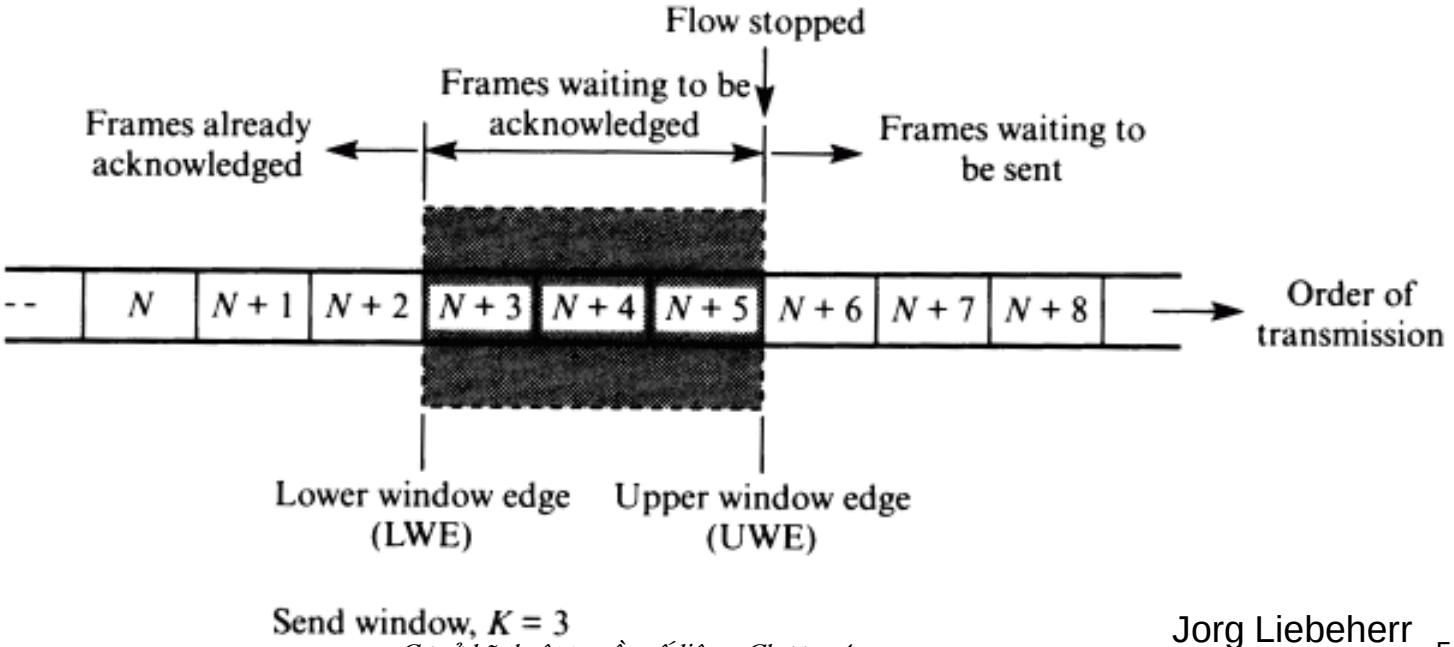


Still used between a computer and a printer

Still used on some half duplex circuits, but it is fading

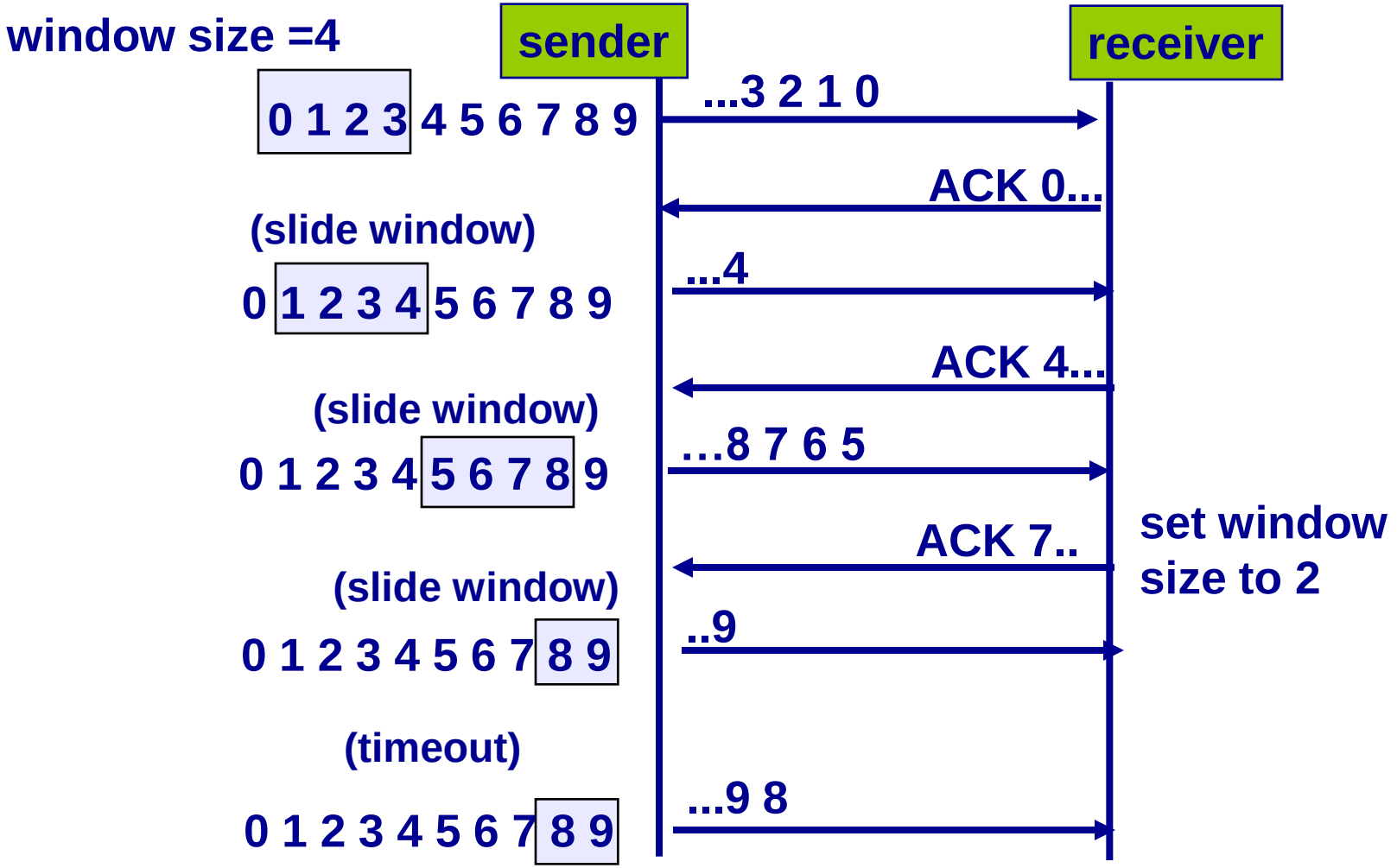
Điều khiển luồng với cửa sổ trượt

- **Sliding Window Flow Control**
  - Allows transmission of multiple frames
  - Assigns each frame an m-bit sequence number
  - Range of sequence number is [0..2^m-1], i.e., frames are counted modulo 2^m

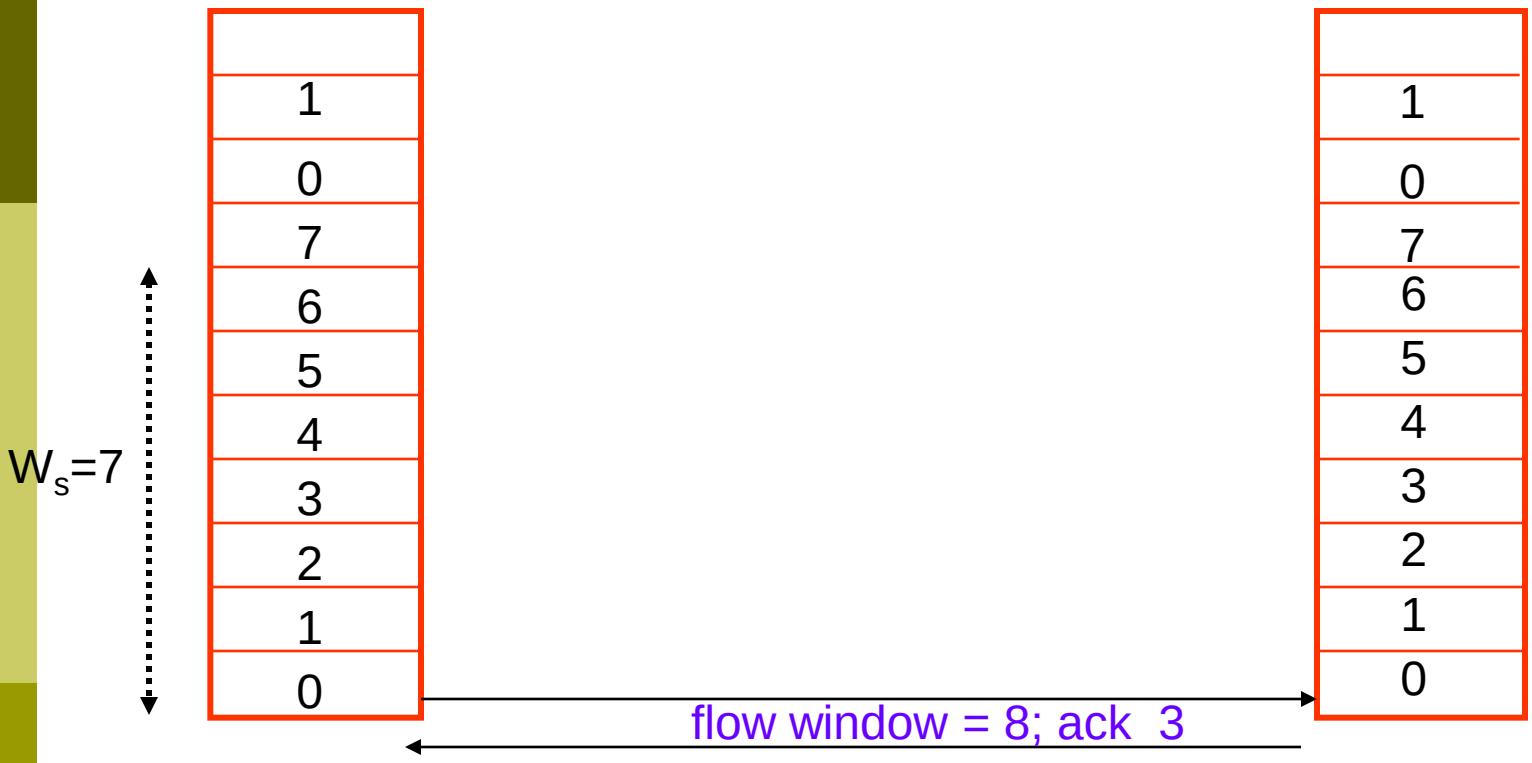


Jorg Liebeherr

Điều khiển luồng với cửa sổ trượt



Sliding window FC illustrated



Sending (retransmission) buffer
Holds frames until acknowledged
If timer times-out, retransmit
(error control)
If m=3, i.e. 3-bit sequence number,
then W_s can be 7

Receiving buffer
(flow control)
(for selective repeat ARQ error control)

Sender sends a minimum of W_s and flow window indicated by the receiver, which in this case is 7; since it has already sent 4 frames (3, 4, 5, 6) that are as yet unacknowledged, it can only send 3 more (seq. numbers, 7, 0 1). 7 is the number of outstanding frames (unack'ed)

Flow window

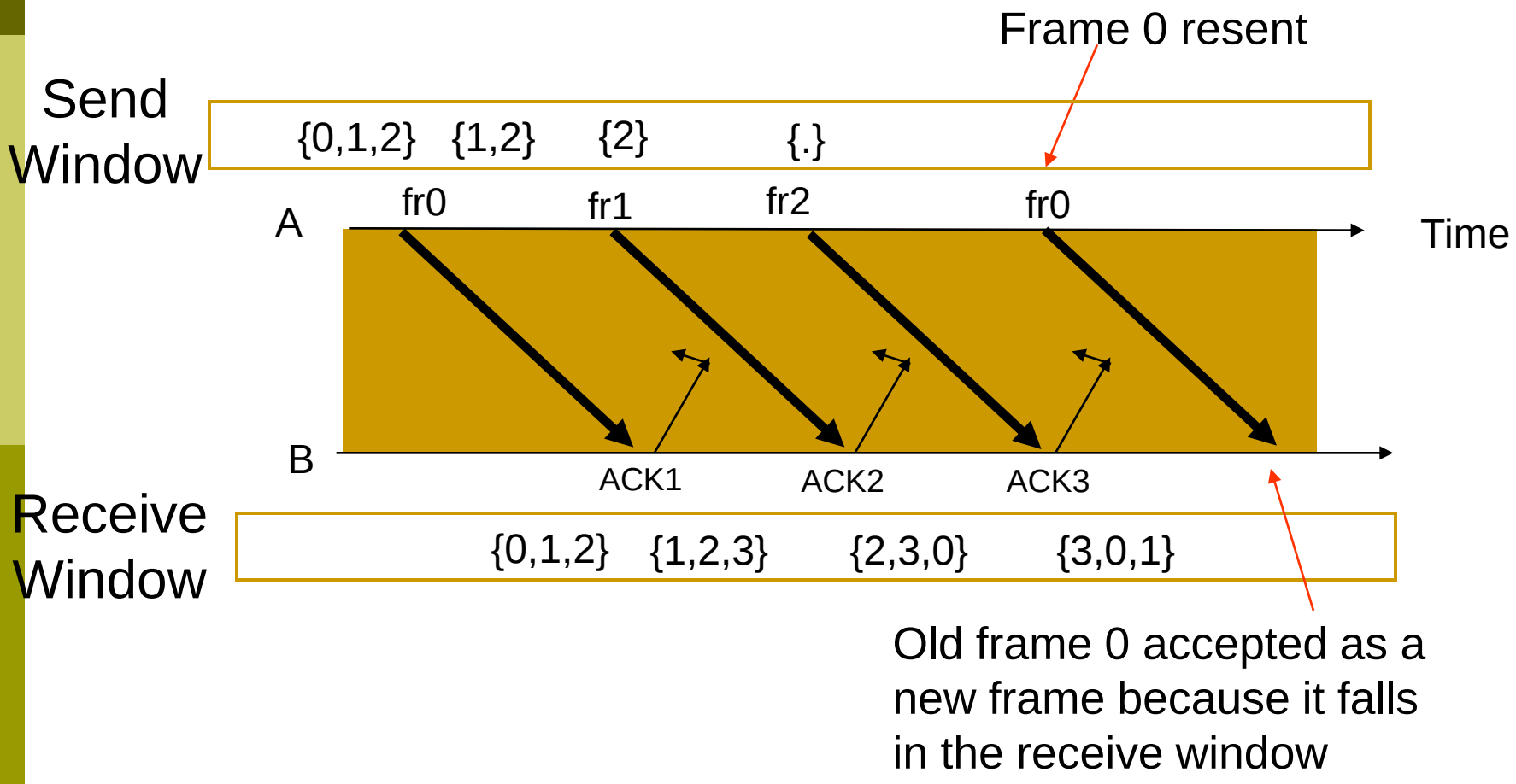
- For sliding flow control:
  - Left-over space in the receiver's buffer
  - In previous slide flow window of 8 is a report of how much space is left in the receiver's buffer (enough to hold 8 frames)
- Flow window also called advertised window or receiver's window
- Receiver buffer has a different purpose in selective repeat ARQ – see next slide

Rules for operation of sliding-window flow control

- At any time instant, the sender can only have outstanding as many frames as the minimum of (W_s, W_r)
- Outstanding – means unacknowledged
- W_s: sending window size (in frames)
- W_r: receiving window size (in frames)
- Note that send buffer size can be larger than the sending window size.
- If running selective repeat ARQ, $W_s + W_r = 2^m$ is maximum allowed (see next three slides) where m is the length in bits of the sequence number field
- If selective repeat ARQ is used in conjunction with sliding flow control, then W_r is a minimum of the two numbers dictated by selective repeat ARQ need for $W_s + W_r = 2^m$ and the sliding window flow control need to indicate the left-over space in the receive buffer

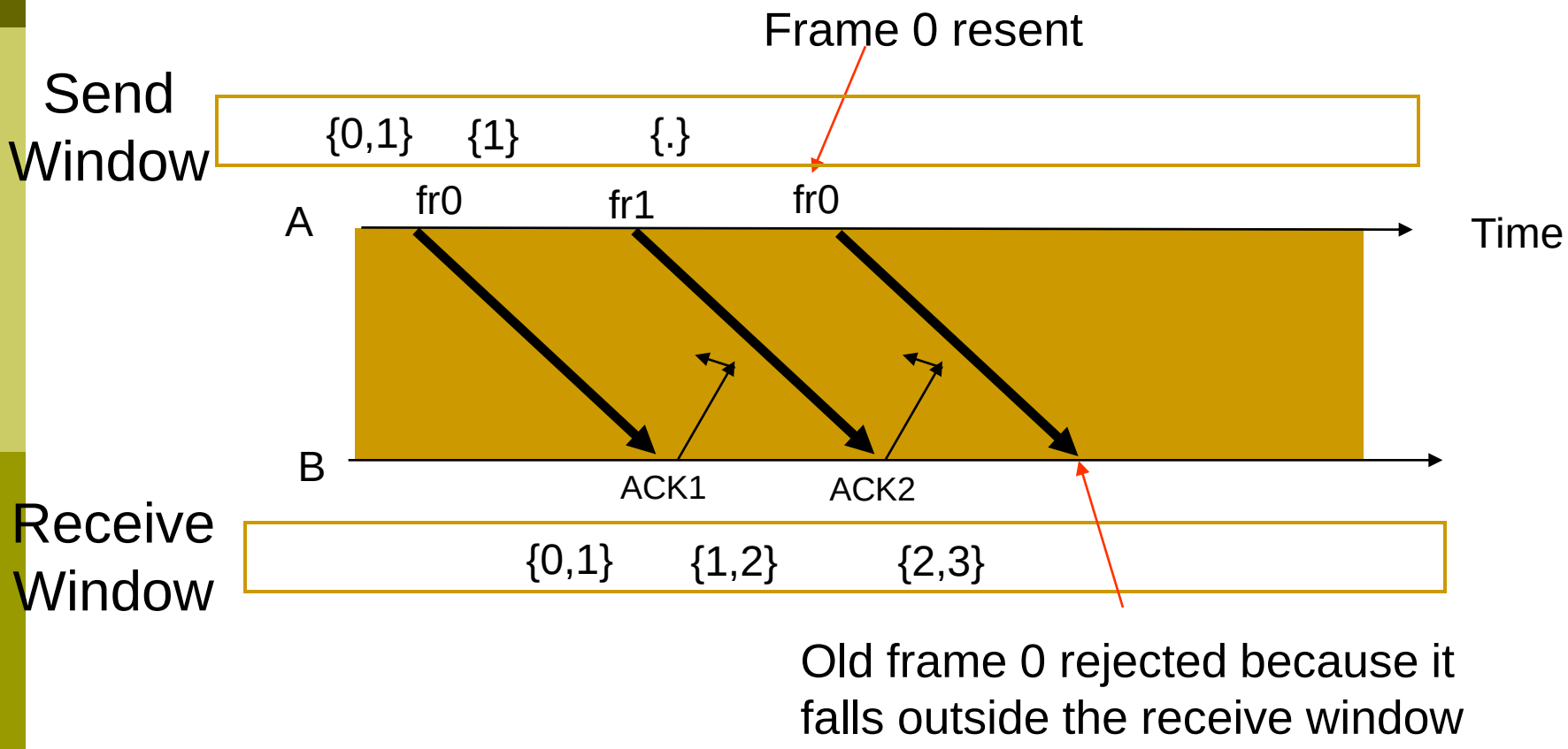
Selective repeat ARQ: What sizes are allowed for W_s and W_r?

- Example: $M=2^2=4$, $W_s=3$, $W_r=3$



Selective repeat ARQ
$W_s + W_r = 2^m$ is maximum allowed

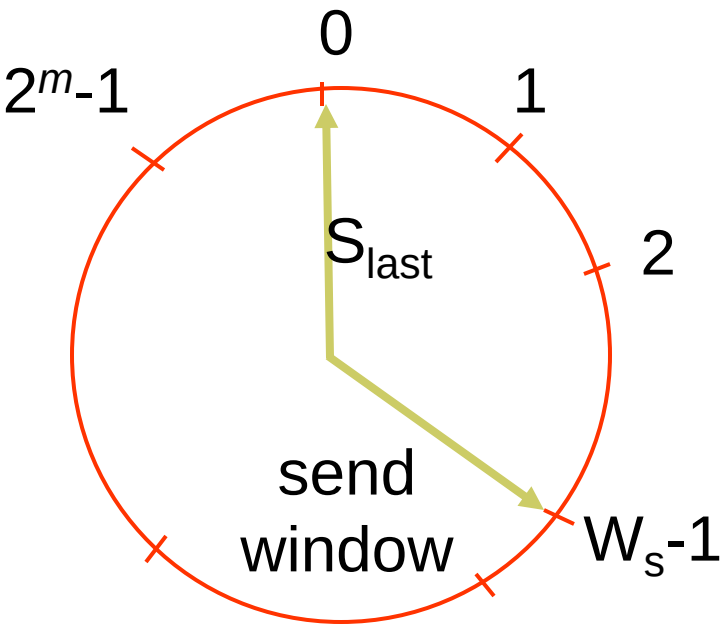
- Example: $M=2^2=4$, $W_s=2$, $W_r=2$

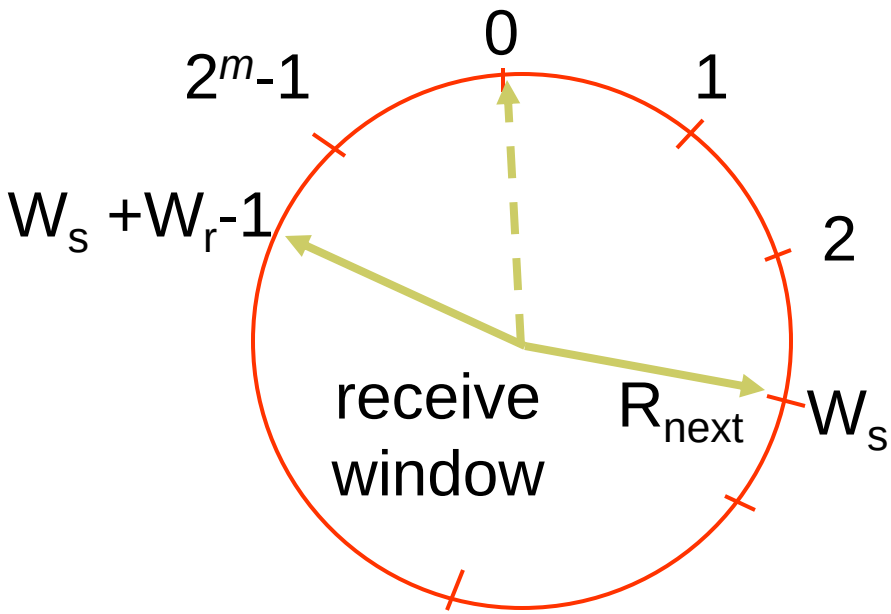


Selective repeat ARQ
Why $W_s + W_r = 2^m$ works

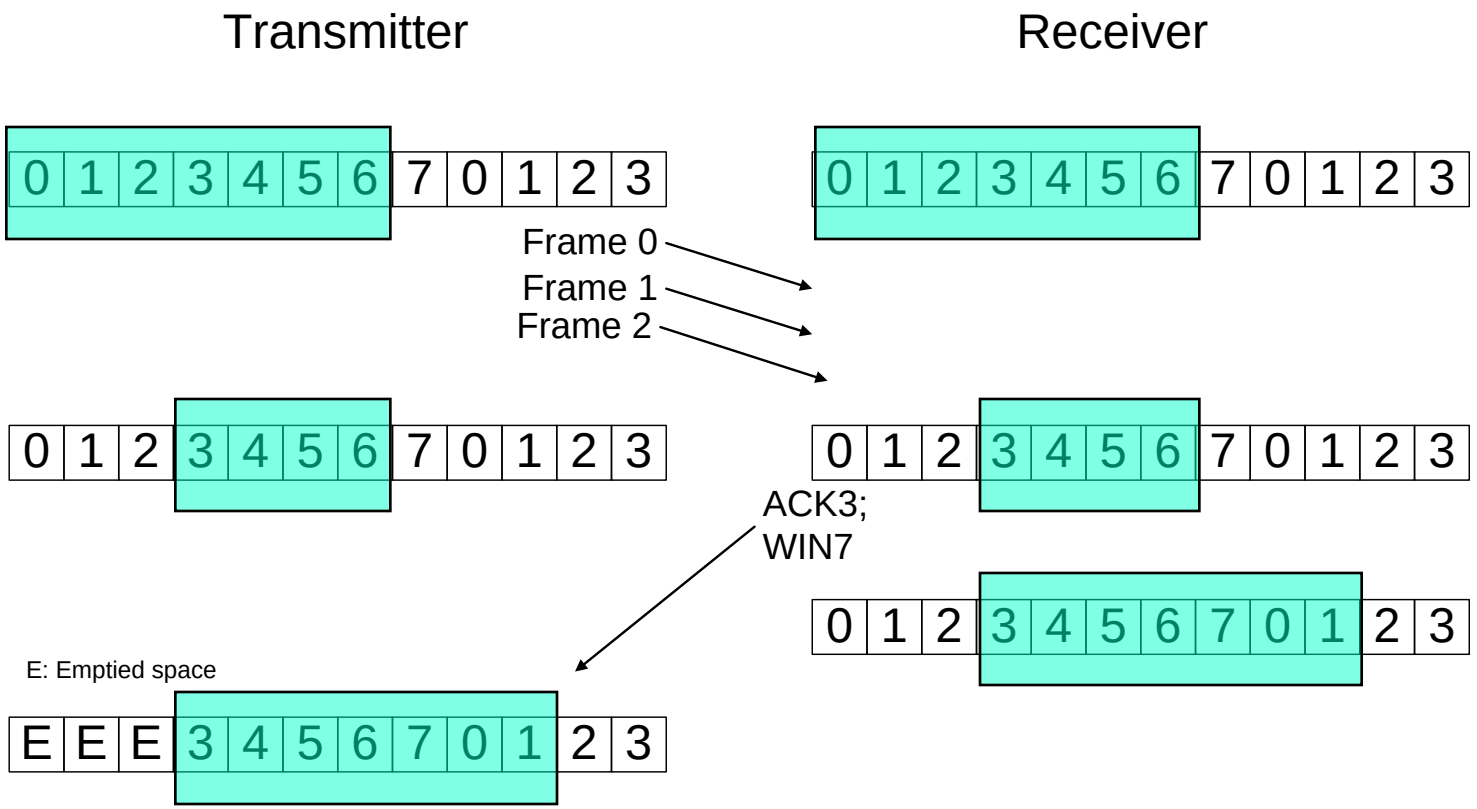
- Transmitter sends frames 0 to Ws-1; send window empty
- All arrive at receiver
- All ACKs lost
- Transmitter resends frame 0

- Receiver window starts at $\{0, \ldots, W_r\}$
- Window slides forward to $\{W_s, \ldots, W_s + W_r - 1\}$
- Receiver rejects frame 0 because it is outside receive window

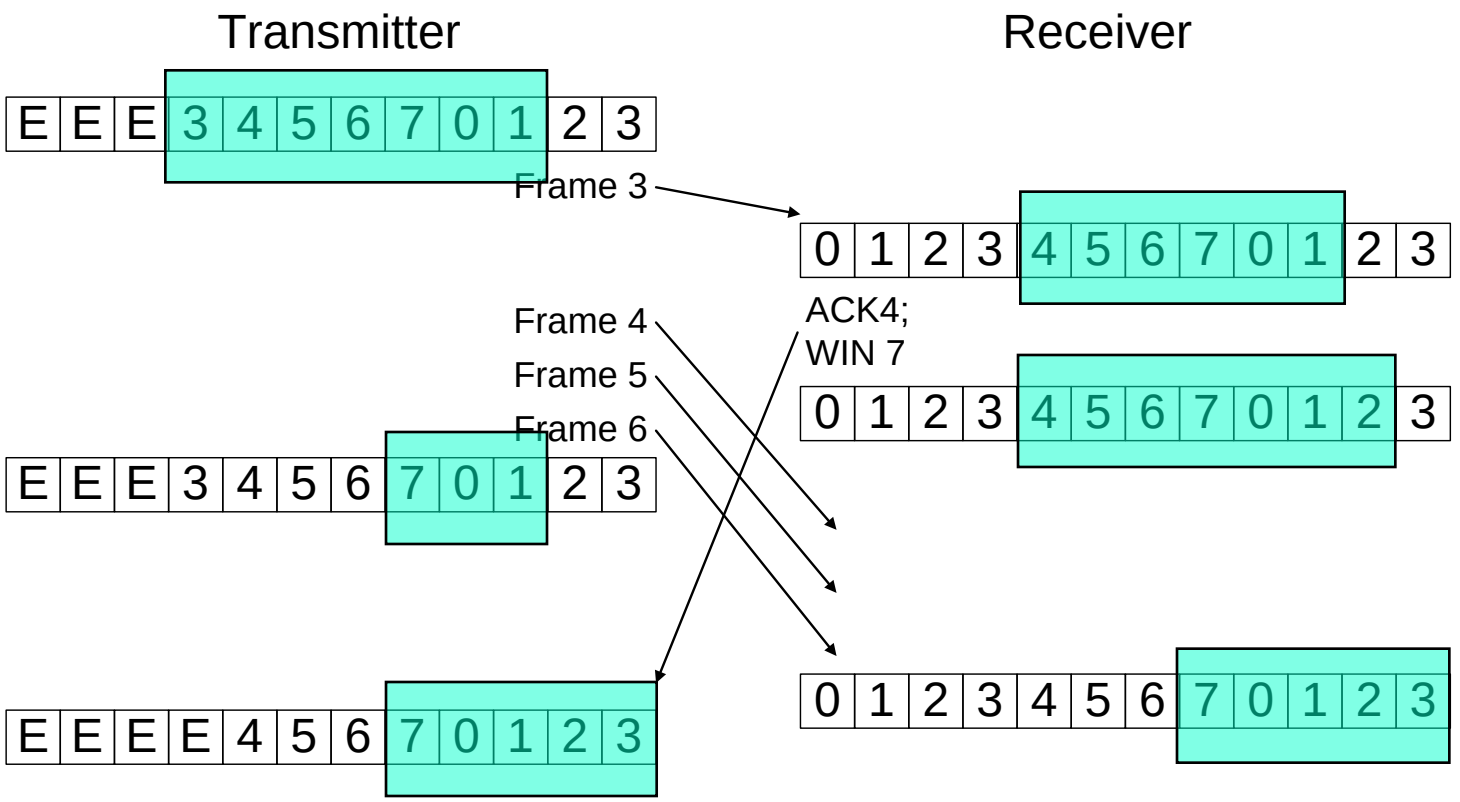




Example of sliding window FC



Example Continued



Interaction of flow control with different ARQ schemes

- **Remember in Go-Back-N ARQ the receive buffer only needs to hold one frame.**
- **But, in selective ARQ, the receiver has a buffer for error correction.**
- **This same buffer can be used for flow control with its left-over space size being reported along with the expected sequence number in ACKs.**
- **If sliding window flow control is used with Go-Back-N ARQ, the receiver needs a buffer**
  - **the sequence number sent in the ACK is R_{next} and all frames from that number will be retransmitted if necessary (Go-Back-N)**
  - **for flow control, the size of the left-over space in the buffer will be carried as a separate parameter in the ACK**

Rate control

- The receiver sends feedback to the sender to allow it to adjust its sending rate
- Example: new TCP algorithms

Communication problems

- corruption of packets
- loss of packets
- replication of packets
- ordering of packets
- independent failures of nodes

Communication protocols

- We have assumed until now that packets cannot be reordered
  - this when the parties are connected by a single physical line
- What if the channel is a network?
  - the packets may arrive out of order
- Sequence number might be reused – duplications possible
- How to solve this problem?
- Introduce the lifetime of the packet

Connection establishment

- Connection request PDU
- Connection accepted PDU

- Is this enough?
  - What if the network can store or duplicate packets?
  - What if the subnet is so congested that ACK or datagrams cannot go through?

Example: bank transaction (delayed duplicates)

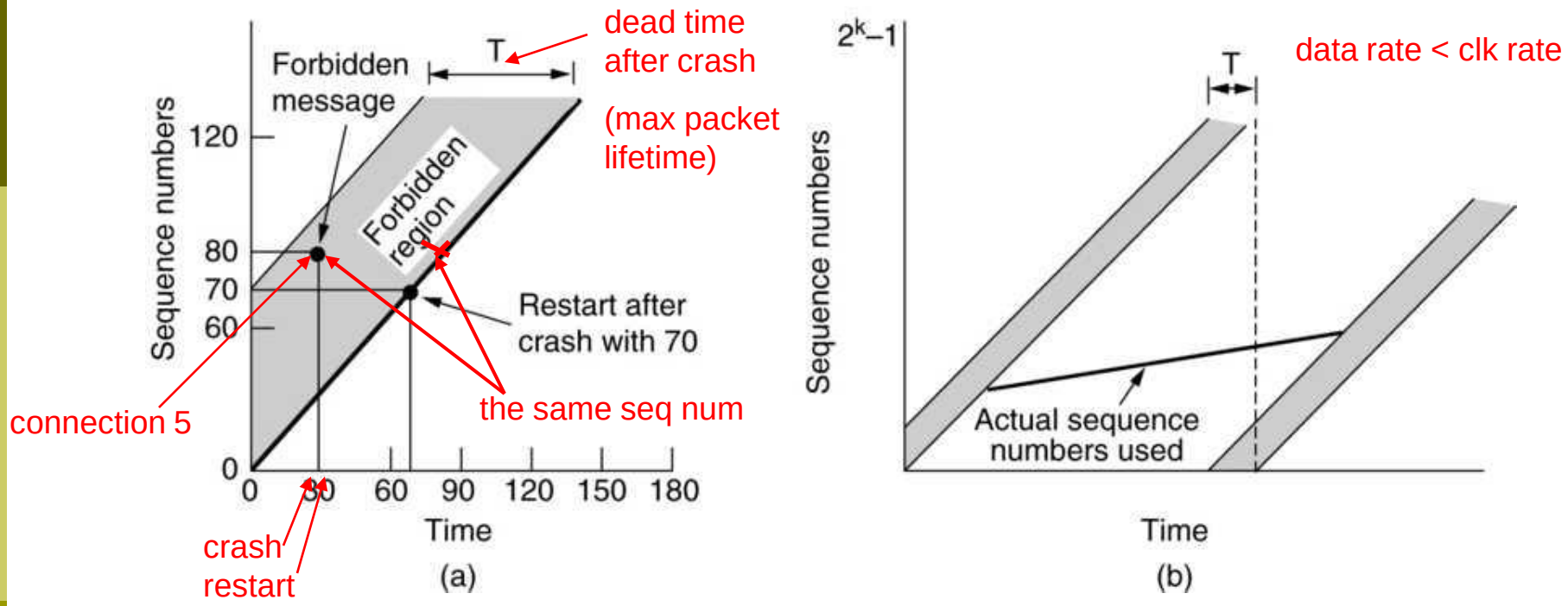
Solutions:

- new addresses for each connection?
- incremented sequence numbers?
- PACKET LIFETIME

Packet lifetime

- Can be restricted using:
  - restricted subnet design
    - prevents looping
    - bounding congestion delay over the longest possible path
  - putting a hop counter in each packet
    - decrementing default value & dropping when zero
  - time stamping each packet
    - carry the time when it was created
    - discarded after some predefined time
      - all the routers synchronized – nontrivial
- Lifetime expires:
  - Not only the packets but all the ACKs must be dead

Connection Establishment (2)

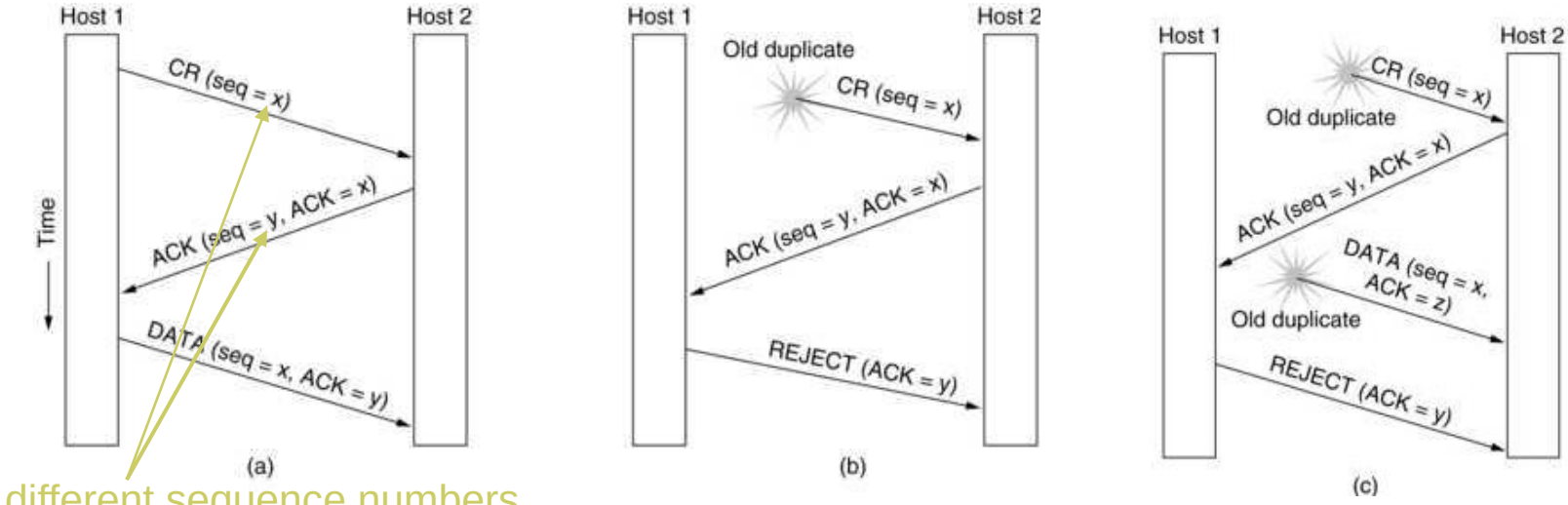


Time-of-day clock (not necessarily sync) – 2 PDU with the same seq. num. are never outstanding at the same time

solves the delayed duplicate problem for data PDU

PDUs may not enter the forbidden region.

Connection Establishment (2)

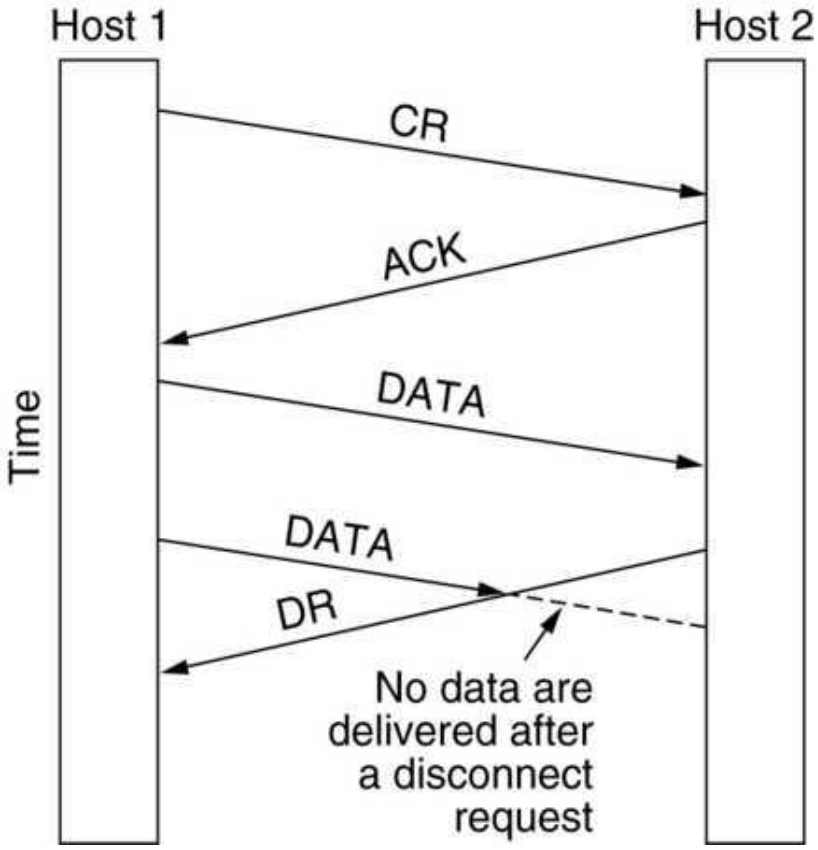


different sequence numbers used by the sender & receiver

Three protocol scenarios for establishing a connection using a three-way handshake. **CR** denotes CONNECTION REQUEST.

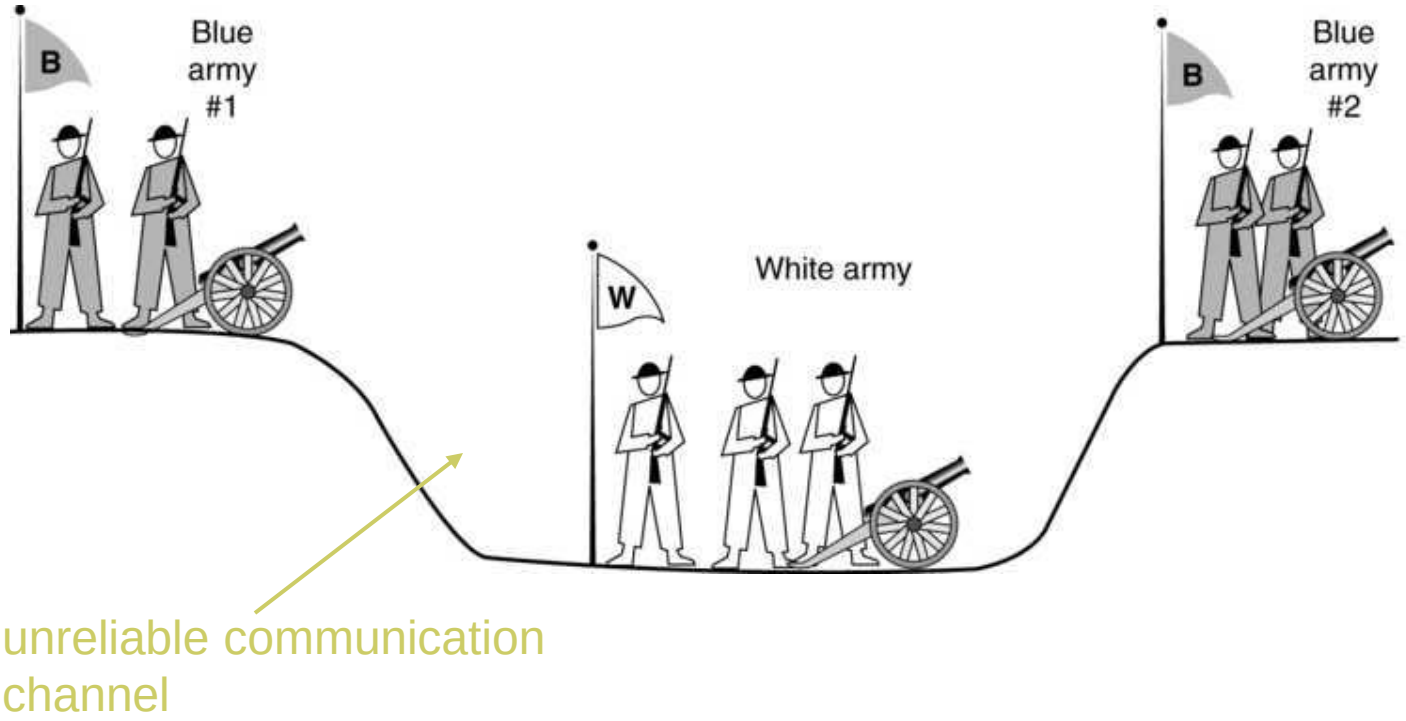
(a) Normal operation,
(b) Old CONNECTION REQUEST appearing out of nowhere.
(c) Duplicate CONNECTION REQUEST and duplicate ACK.

Connection Release



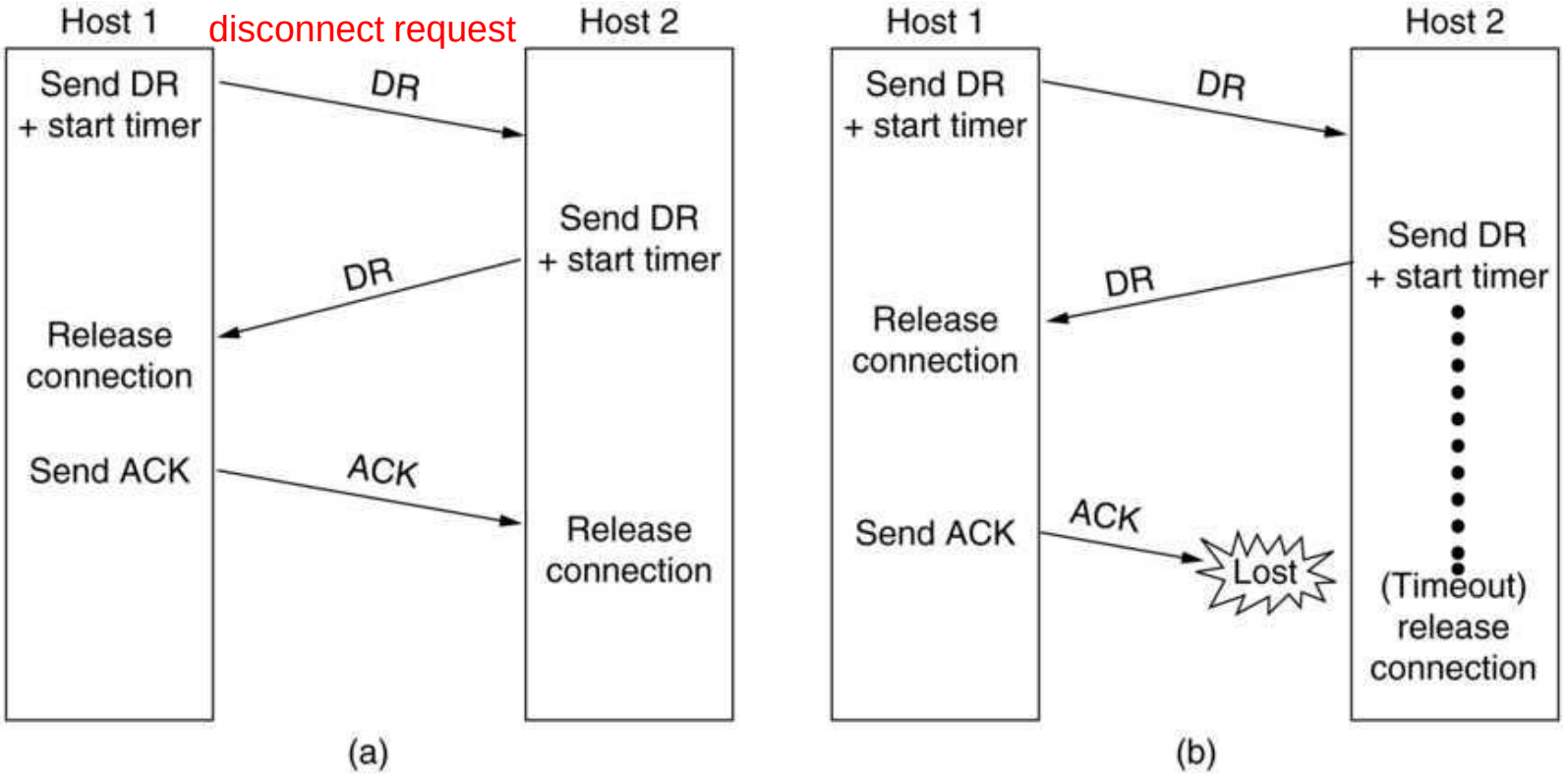
- asymmetric release
  - a telephone line
    - connection is broken when one party hangs up
  - abrupt disconnection with loss of data.
- symmetric release
  - each connection released separately
  - continue to receive after disconnect PDU is sent
  - avoid data loss

Connection Release (2)



The two-army problem.

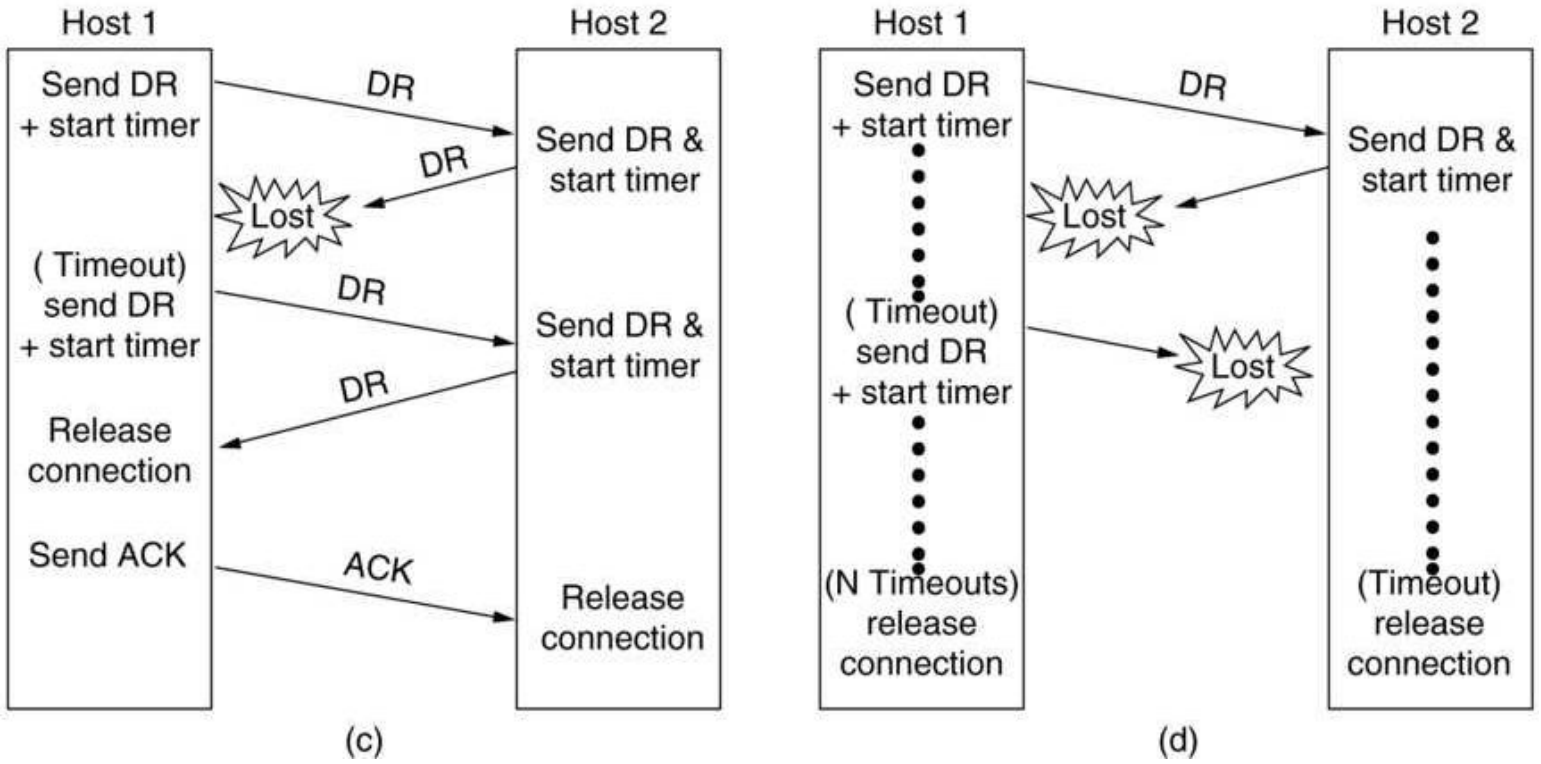
Connection Release (3)



Four protocol scenarios for releasing a connection.

(a) Normal case of a three-way handshake. (b) final ACK lost.

Connection Release (4)



(c) Response lost. (d) Response lost and subsequent DRs lost.