Data Link Layer

José Augusto Afonso
jose.afonso@dei.uminho.pt

Data Link Layer

- Objectives:
  - Achieving reliable communication between two adjacent machines
  - Sharing the medium among different machines
- Design Issues:
  - Framing: data are sent in blocks called frames, the beginning and end of each frame must be recognized by the receiver
  - Error control: bit errors introduced by the transmission system should be detected and/or corrected
  - Flow control: the sending station must not send frames at a rate faster than the receiving station can absorb them
  - Addressing: on a multipoint line, such as a LAN, the identity of the two stations involved in a transmission must be specified
  - Transmit control information and data on the same line
  - MAC - Medium access control (will be discussed later)

Framing

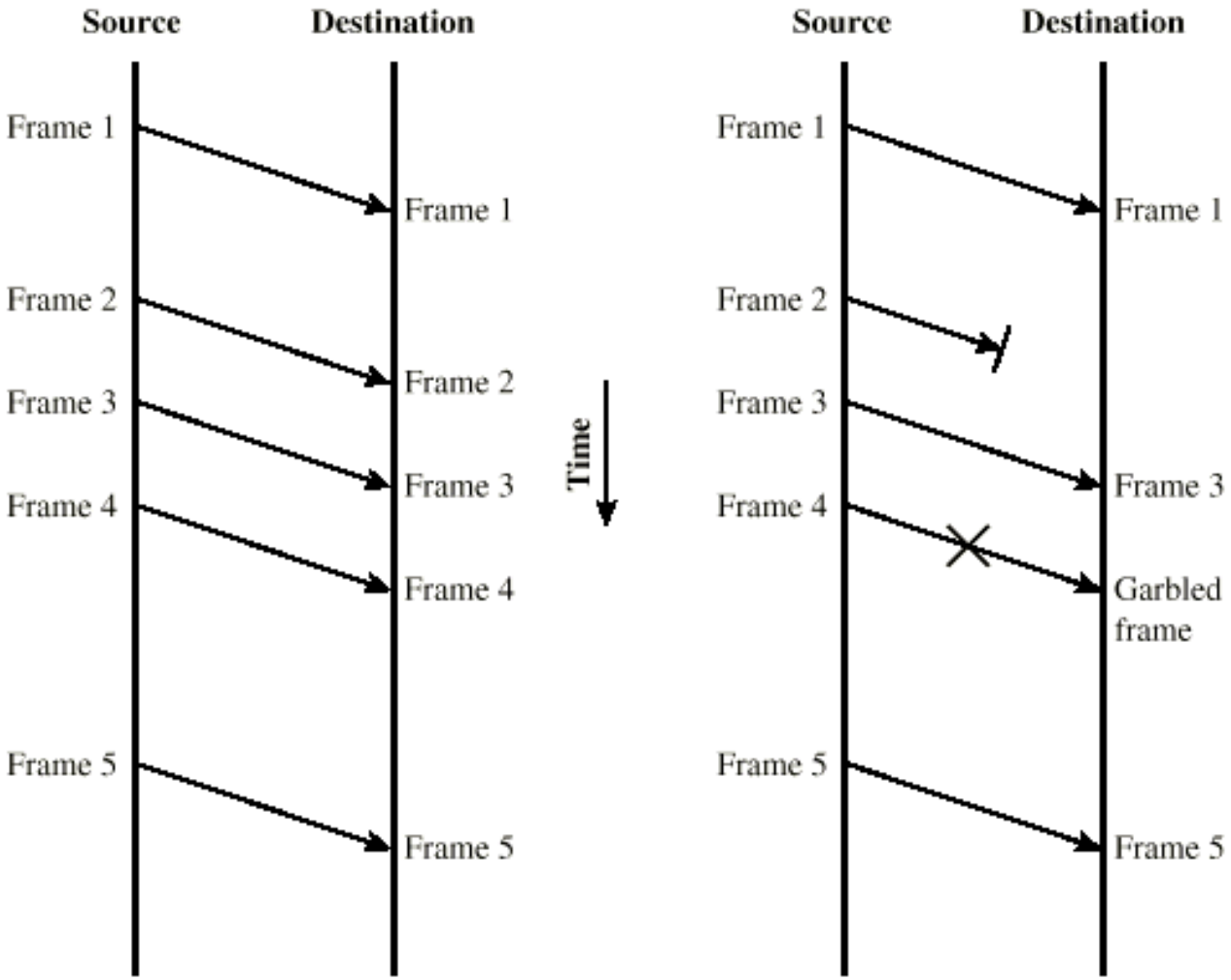
- Identification of the start and end of the frame
  - Uses start/end flag bytes and length fields
- Large blocks of data are broken up into small frames at the source because:
  - limited buffer size at the receiver
  - A larger block of data has higher probability of error
    - With smaller frames, errors are detected sooner, and only a smaller amount of data needs to be retransmitted
  - On a shared medium, such as Ethernet and Wireless LAN, small frame size can prevent one station from occupying medium for long periods

3

Flow Control

- Ensuring the sending entity does not overwhelm the receiving entity
  - Preventing buffer overflow
- Transmission time (T_{TX})
  - Time taken to emit all bits into medium at the sender's side
  - Determined by the data rate
- Propagation time (T_p)
  - Time for a bit to traverse the link and reach the destination
  - Determined by the transmission distance
- We first assume error-free transmission.

Model of Frame Transmission



(a) Error-free transmission

(b) Transmission with losses and errors

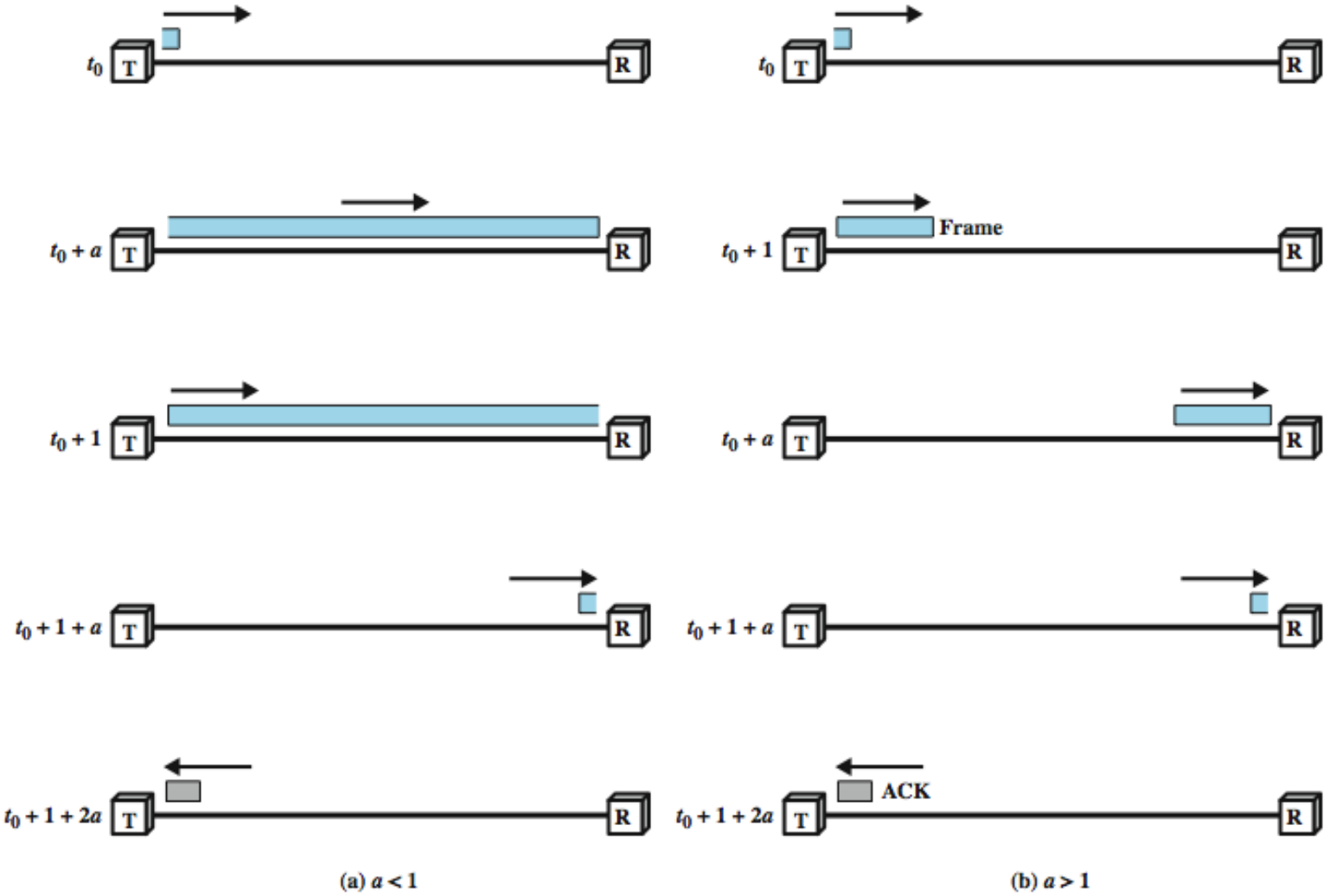
Stop and Wait Flow Control

- Source transmits a frame
- Destination receives the frame, and replies with a small frame called acknowledgement (ACK)
- Source waits for the ACK before sending the next frame
- Destination can stop the flow by not sending ACK (e.g., if the destination is busy …)

Performance of Stop and Wait

- Assumptions
  - Transmission time of the data frame is 1
  - Transmission time of the ACK frame is 0
  - Propagation time is a
    - **a is the ratio of propagation time (T_p) over transmission time (T_{TX})**
  - Error-free transmission
- **The efficiency (channel utilization ratio) is $U = 1/(1+2a)$**
  - In a time period of $1+2a$, the transmitter is only busy with 1 unit of time.
- It is not efficient for long haul transmission and high speed transmission.
  - Another type of protocol called “sliding window” is designed for this situation.

Stop and Wait Channel Utilization



Sliding Window Flow Control

- Idea: allow multiple frames to be transmitted before reception of acknowledgment
  - Receiver has a buffer of *W* frames
  - Transmitter can send up to *W* frames without receiving ACK
- Each frame needs to be numbered: sequence number is included in the frame header
  - Sequence number is bounded by the length of "sequence number field" in the header, e.g., *k* bits
  - Frames are numbered modulo 2^k
- ACK includes the sequence number of the next expected frame by the receiver

Performance of Sliding Window

- Assumptions
  - Window size is W
  - Frame transmission time is 1
  - ACK transmission time is 0
  - Propagation time is a
  - Error-free transmission
- The channel utilization ratio is

$$U = \begin{cases} 1 & W \geq 2a+1 \\ \dfrac{W}{2a+1} & W < 2a+1 \end{cases}$$

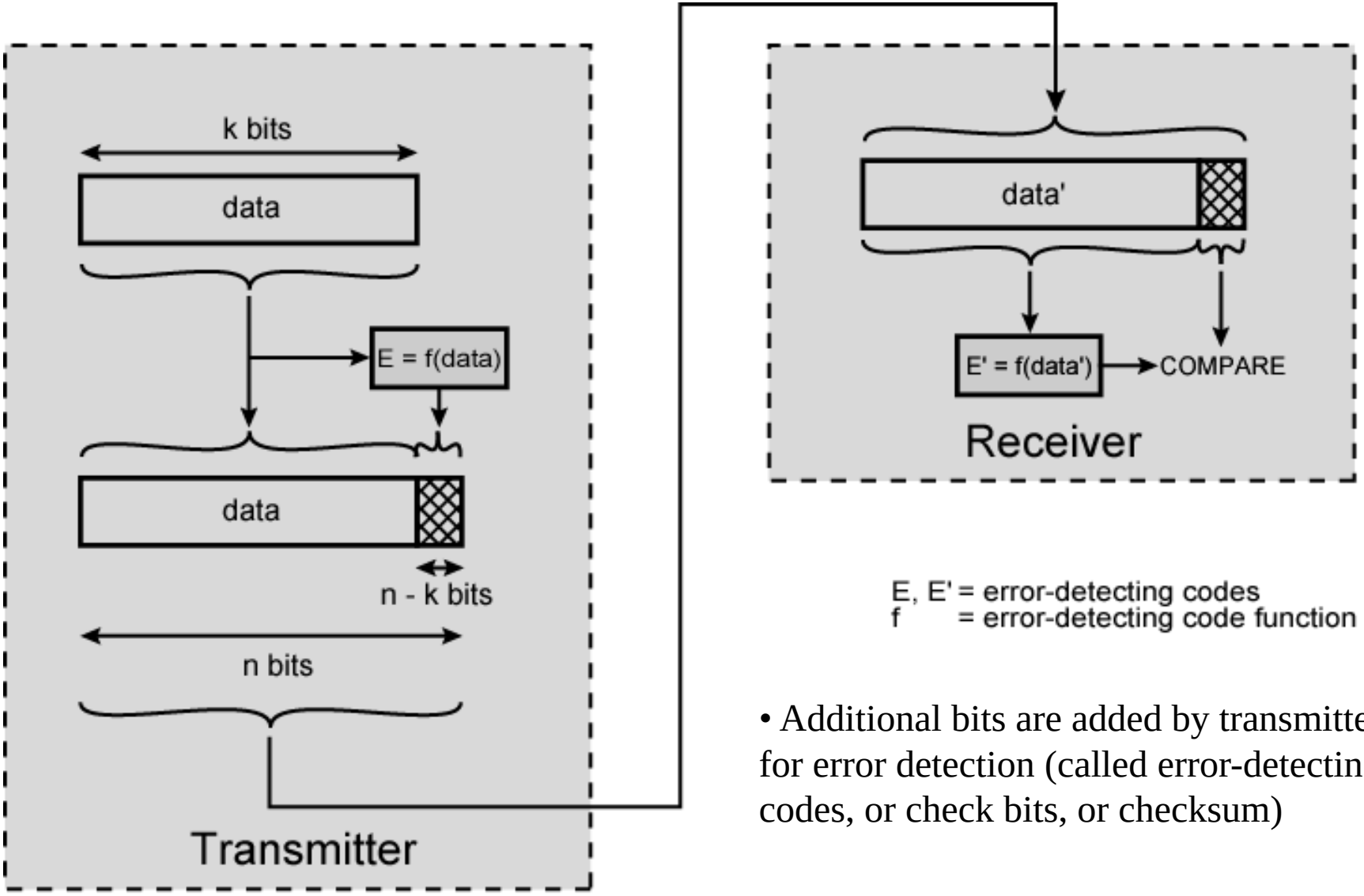
Error Control in Link Layer

- Error control: **detection** and **correction** of errors
- We consider two types of errors:
  - Lost frames
    - The receiver cannot recognize that this is a frame
  - Damaged frames
    - The receiver can recognize the frame, but some bits are in error
- Two approaches for error control
  - ARQ: automatic repeat request, based on some or all of the following ingredients:
    - Error detection
    - Positive acknowledgment
    - Retransmission after timeout
    - Negative acknowledgement and retransmission
  - FEC: forward error correction

Error Detection: Types of Error

- An error occurs when a bit is altered between transmission and reception
- Single bit errors
  - Errors are spread uniformly
  - One bit is altered with probability P_e
  - Adjacent bits are normally not affected
  - Can occur in the presence of white noise (thermal noise)
- Burst errors
  - A cluster of bits with Length B is affected
  - First, last and a number of intermediate bits in error (not necessarily all the bits in the cluster suffer an error)
  - More common and more difficult to deal with
  - Can be caused by impulse noise
  - Common in wireless links

Error Detection Process



E, E' = error-detecting codes
f = error-detecting code function

• Additional bits are added by transmitter for error detection (called error-detecting codes, or check bits, or checksum)

Parity Check

- Append a parity bit to the end of a block of data
- Value of parity bit is such that the new data has even (even parity) or odd (odd parity) number of ones
  - E.g., original data 1110001 -> 11100011 (odd parity)
- Limitation: even number of bit errors goes undetected
  - E.g., 11100011 -> 11010011 (undetected!)
- Used by RS-232 protocol

14

Cyclic Redundancy Check

- For a block of *k* bits, transmitter generates an (*n*-*k*)-bit sequence called Frame Check Sequence (FCS)
- The resulting frame consisting of *n* bits is exactly divisible by some predetermined number.
- Receiver divides the incoming frame by the predetermined number:
  - If no remainder, assume no error

15

Automatic Repeat Request (ARQ)

- The effect of ARQ is to turn an unreliable data link into a reliable one.
- Three versions of ARQ:
  - Stop-and-Wait
  - Go-back-N
  - Selective repeat (or selective reject)

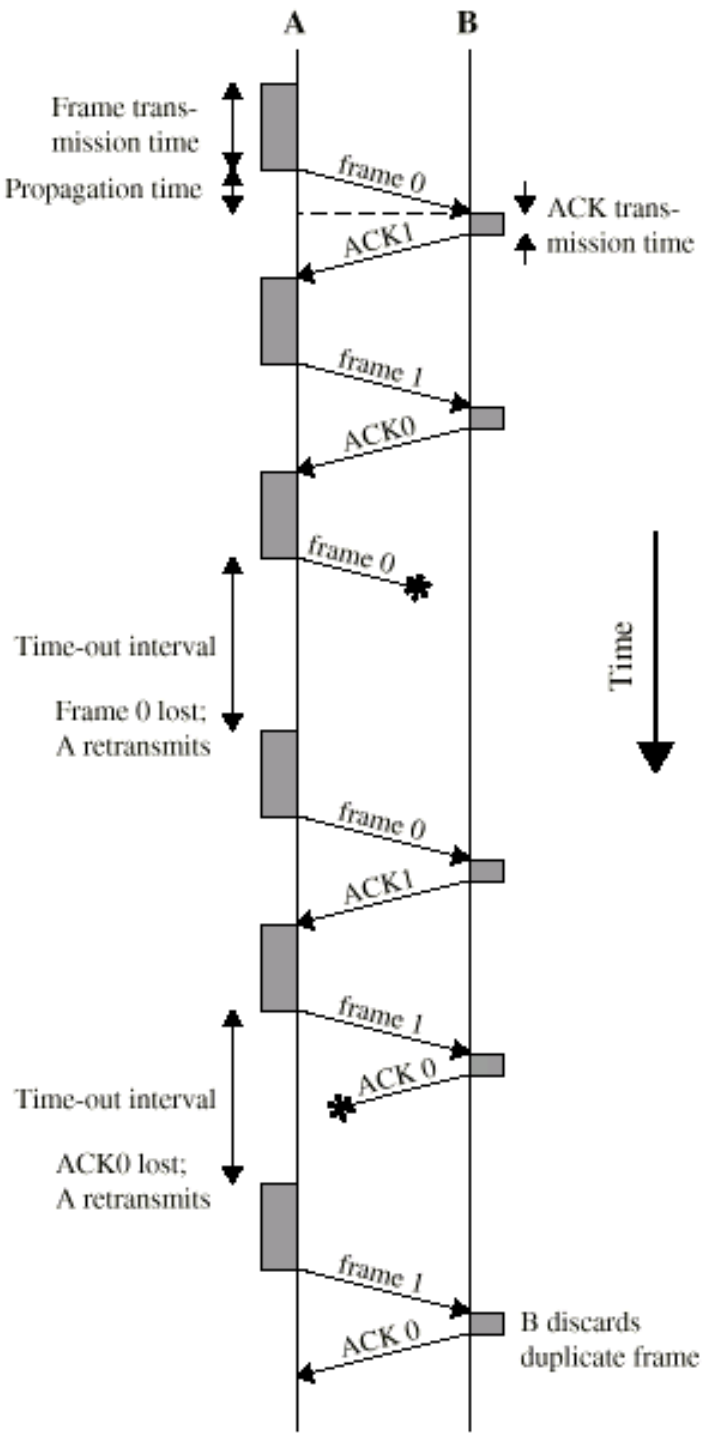
Stop and Wait ARQ

- Based on stop and wait flow control
- The source station is equipped with a timer
- Source transmits a single frame, and waits for an ACK
- If the frame is lost…
  - The timer eventually fires, and the source retransmits the frame
- If receiver receives a damaged frame...
  - The receiver discards it, the source timer eventually fires, and the source retransmits the frame
  - Or the receiver transmits a NACK (negative acknowledgment)
- If everything goes right, but the ACK is damaged or lost, the source will not recognize it
  - The timer eventually fires, the source will retransmit the frame
  - Receiver gets two copies of the same frame!
  - Solution: use sequence numbers, 1 bit is enough, i.e., frame0 and frame1, ACK0 and ACK1

Stop-and-Wait Diagram

Simple, but inefficient for long distance and high speed applications.

We can use sliding window technique to improve the efficiency.

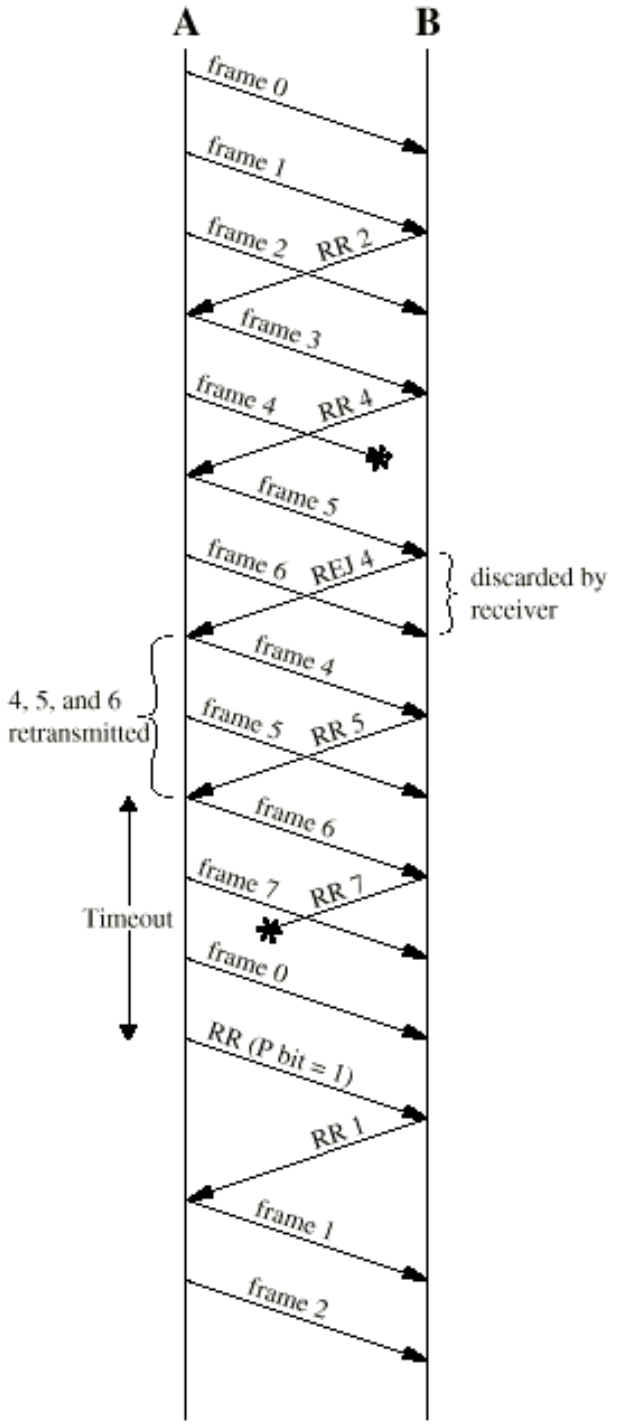


Go-Back-N ARQ

- Based on sliding window flow control
- Use window size to control the number of unacknowledged frames outstanding
- If no error, the destination will send ACK as usual with next frame expected (positive ACK, RR: receive ready)
- If error, the destination will reply with rejection (negative ACK, REJ: reject)
  - **Receiver discards that frame and all future frames, until the erroneous frame is received correctly**
  - **Source must go back and retransmit that frame and all succeeding frames that were transmitted in the interim**
  - This makes the receiver simple, but decreases the efficiency

Go-Back-N Diagram

For a k-bit sequence number, the window size can be at most 2^k-1, otherwise RR 0 is ambiguous (e.g., first sends frame 0 and gets back an RR1, and then sends frames 1,…,7,0, and gets another RR1).



Selective Reject ARQ

- Also called selective repeat
- Pros:
  - Only rejected frames are retransmitted
  - Subsequent frames are accepted by the receiver and buffered
  - Minimizes the amount of retransmissions
- Cons:
  - Receiver must maintain large enough buffer, and must contain logic for reinserting the retransmitted frame in the proper sequence
  - Also more complex logic in the source

Selective Reject Diagram

For a k-bit sequence number, the window size can be at most 2^{k-1}, because the sending and receiving windows overlap.

Assume k=3, and window size is 5.

1. A sends frames 0, 1, …, 4 to B.

2. B receives all 5 frames, and cumulatively acknowledges with RR5.

3. RR5 is lost.

4. A times out, and retransmits frame 0.

5. B is expecting a new set of frames 5, 6, 7, 0, 1. So it will accept the retransmitted frame 0 and regard it as a new frame, which is wrong.