Dr Takfarinas SABER
takfarinas.saber@dcu.ie

CA169
Networks & Internet

Link Layer
Part 3: Flow Control

Flow and Error Control

- **Error control** allows the Receiver to tell the Sender about frames damaged or lost during transmission
  - using acknowledgements (ACK)
  - coordinates the *re-transmission* of those frames by the Sender

- **Flow control** specifies how much data the Sender can transmit before receiving *permission to continue* from the Receiver

- Flow control is responsible of delivering the Receiver's ACK when a frame is received frames,
  - Therefore, flow control is closely linked to error control

Basic Idea of Flow Control

- If the Sender transmits frames faster than the amount of frames that the Receiver can process:
  - the receiver will be forced to drop some of them
  - even if the frames are received without any error

- The Receiver needs to **signal** the Sender to slow down to an acceptable rate for the receiver

- This signal can be **explicit** or **implicit** (e.g. delay sending ACK to Sender)

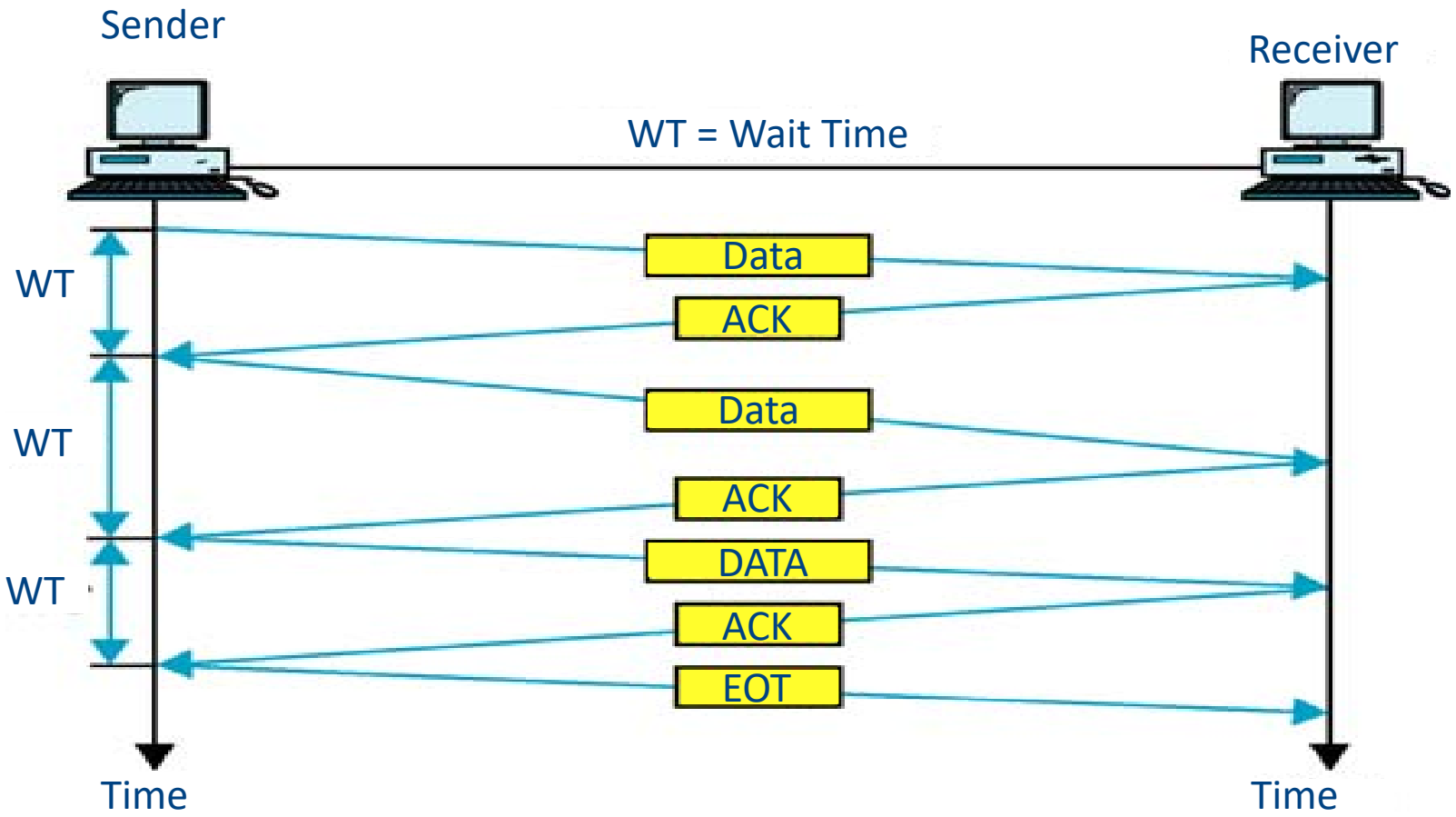
Basic Idea of Error Control

- ACK every correctly-received frame and negatively acknowledge (NAK) each incorrectly-received frame
- Sender keeps copies of un-ACKed Frames
  - re-transmits them, if required
- We want packets (inside frames) passed to receiver's network layer **in order**

Stop-and-Wait Flow Control

- In this scheme the Sender waits for ACK after each frame is transmitted
- An ACK can be either:
  - a frame by itself
  - or a control field in data frames going from receiver to sender (piggybacking)
- This scheme is very simple but also is very inefficient because of the wait times

Stop-and-Wait Flow Control

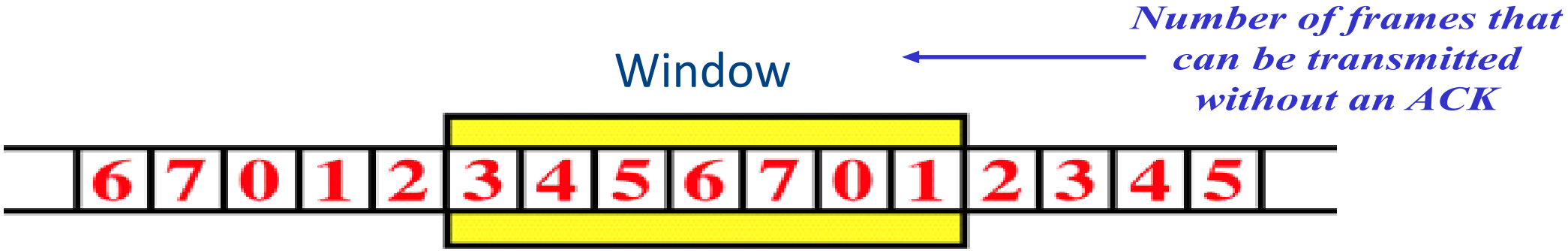


How can we improve on this?

Sliding Window Flow Control

- Sender can transmit several frames **continuously** before needing an ACK
- If ACK is received by the Sender before continuous transmission is finished
  - the Sender can continue transmitting
- An ACK can acknowledge the correct receipt of **multiple** frames by the Receiver
- Frames and ACKs must be numbered:
  - Each Frame's number is 1 greater than the previous frame
  - Each ACK's number is the number of the ***next frame*** *expected* by the Receiver

Sliding Window Flow Control



Sliding Window Flow Control

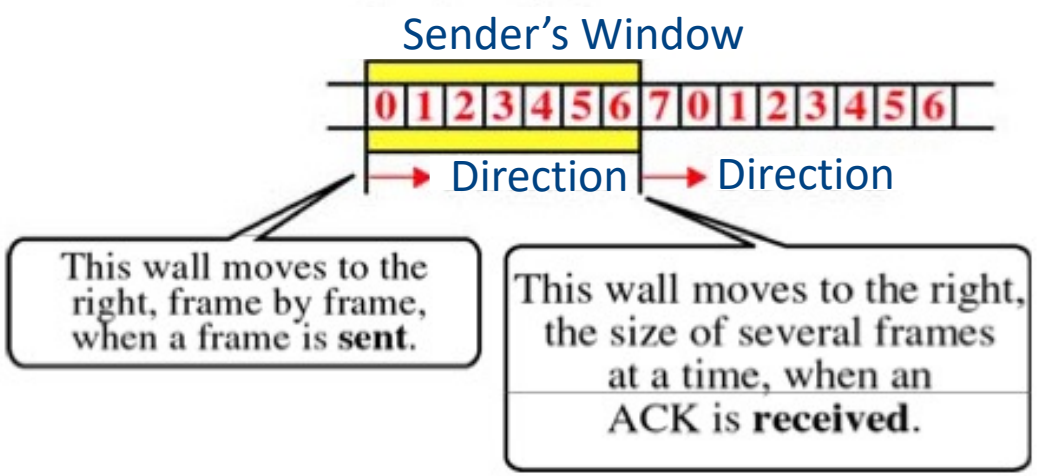
1. Frames may be acknowledged by the Receiver at any time,
2. and Frames may be transmitted by the Sender as long as the **window** hasn't filled up

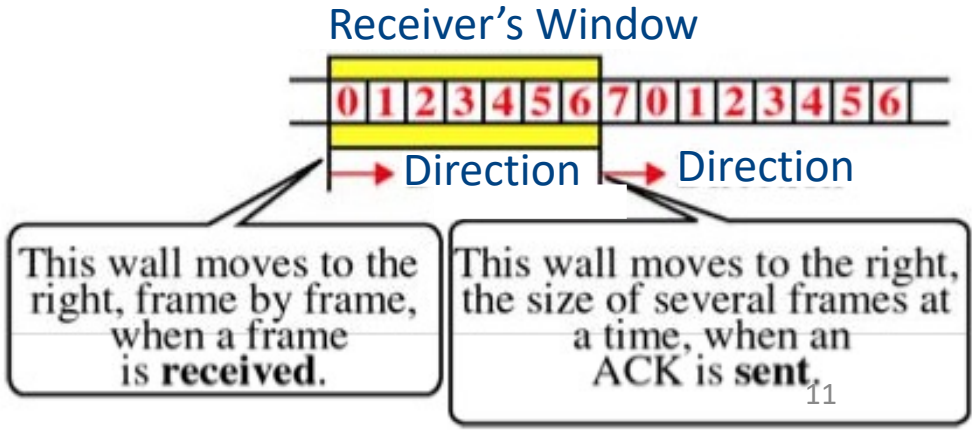
- Frames are numbered modulo-n (i.e., from 0 to n-1):
- **0, 1, 2, …, n-1, 0, 1, 2,…, n-1, 0, 1, 2, …**

- Size of the Window is n-1:
  - 1 less than the number of different Frame numbers

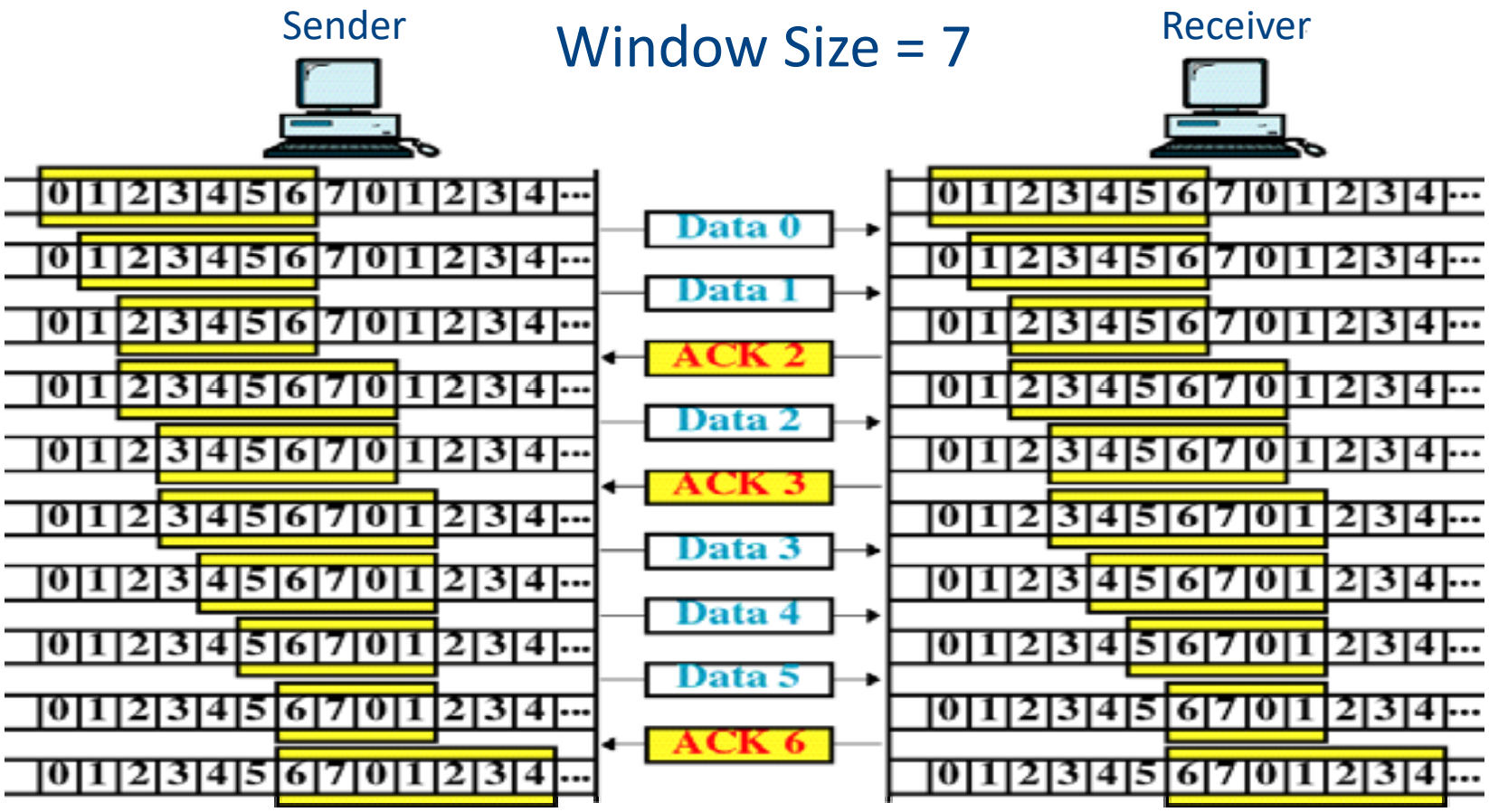
Sliding Window Flow Control

- Both the Sender and Receiver keep track of the sliding window position
- If the Sender receives an ACK with the number 4, then it knows all the frames **up to and including** frame 3 were correctly received
- Receiver's window represents the number of un-ACKed frames





Sliding Window Flow Control Example



What Window Size to Use?

- What is the best window size?
- We do not want to wait to send a message
  - but we do not want to overload the Receiver either.
- The window allows the Sender to “fill a pipe” full of frames …
  - …. Before it expects an ACK form the receiver!

Frame Numbers

- We have to number the frame that is sent
- If the frame CRC is received and it is correct, then send an Acknowledgement (ACK) back to the transmitter
- If the CRC has a remainder then we can send back to the transmitter a Negative ACK (NACK)
- If we miss a frame then the timer kicks in
- The sequence numbers are put in the control field which also tells us which type of frame we have
  - N(S) sent frame number
  - N(R) next requested frame

Dealing with Transmission Errors

- It is hard to send all frames without any errors
  - Even in networks with low Bit Error Rates (BER)

- How would our scheme deal with transmission errors?
  - Make the Sender aware of the error
  - Retransmit the frame
- Automatic Repeat Request (ARQ) Schemes

Automatic Repeat Request (ARQ) Schemes

- If **error detected** in received frame, return NAK to Sender
  - **Explicit:** send a NAK frame to the Sender
  - **Implicit:** Wait until Sender's Timeout timer expires

- Sender keeps a copy of every non ACKed frame to re-transmit if required
  - **Explicit**:
    - ACK received by sender for frame → discard copy
    - NAK received by sender for frame → re-transmit frame
  - **Implicit:**
    - Sender starts timeout timer for each frame (appropriate Timeout value is the expected delay for sender to receive ACK for the frame)
    - ACK received by sender for frame → discard copy
    - Timeout value exceeded → re-transmit frame

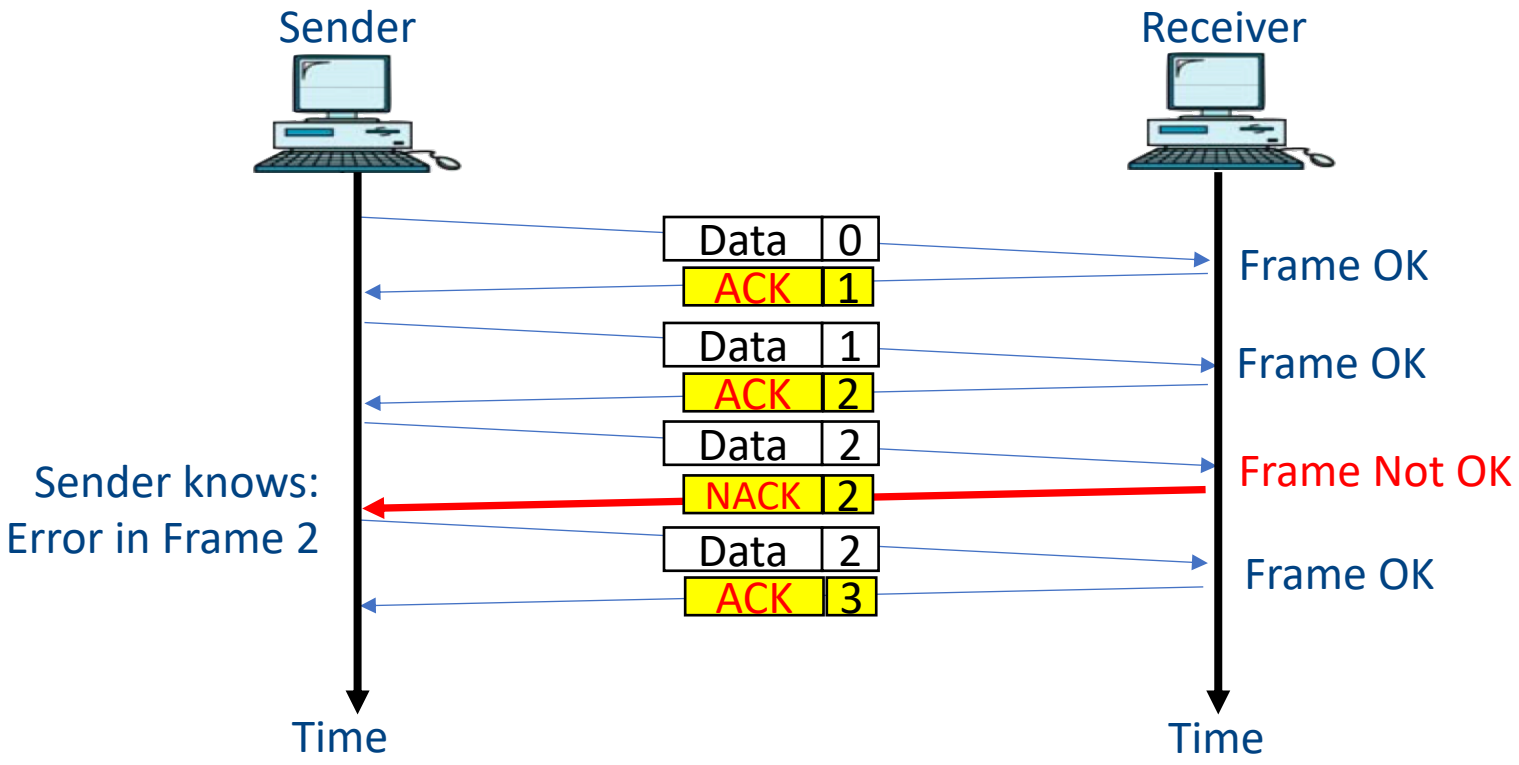
Automatic Repeat Request (ARQ) Schemes

- There are three types of ARQ scheme:

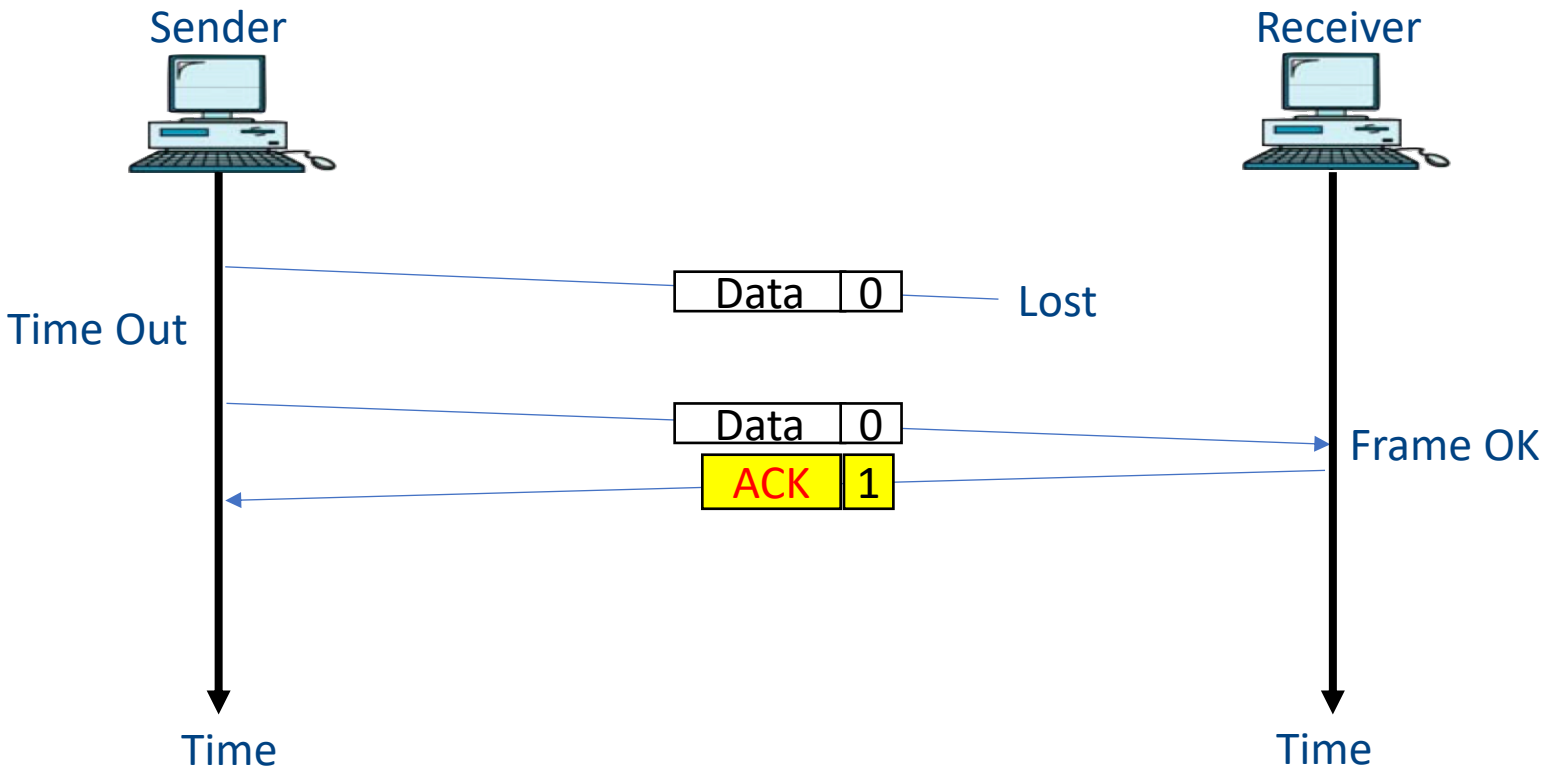
1. ***Stop-and-Wait ARQ:*** extension of Stop-and-Wait flow control

2. **Sliding window ARQ**: extension of sliding window flow control:
   a) ***Go-Back-N ARQ:*** Receiver must get Frames in correct order
   b) ***Selective Repeat ARQ:*** correctly-received out-of-order Frames are stored at Receiver until they can be re-assembled into correct order

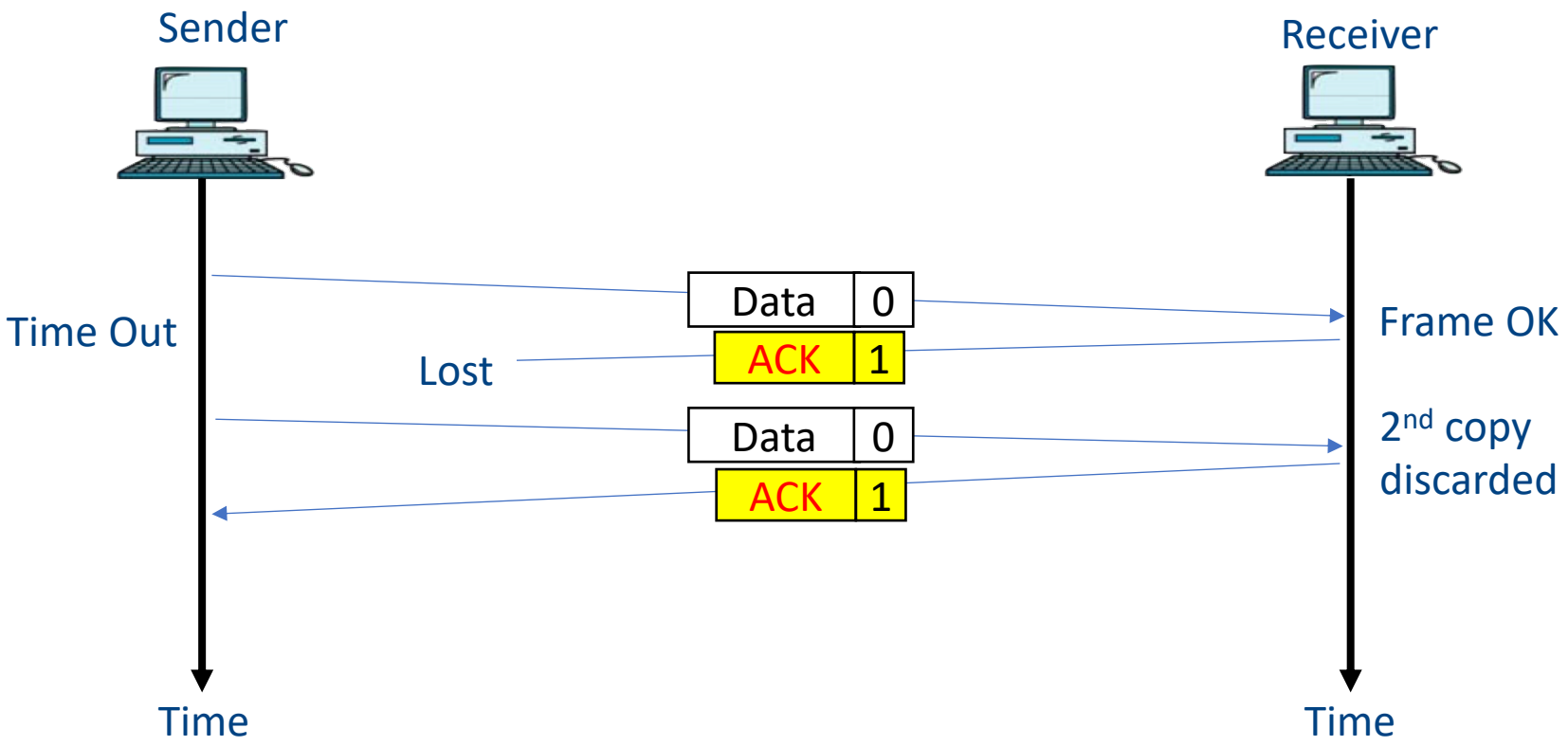
Stop-and-Wait ARQ Damaged Frame



Stop-and-Wait ARQ Lost Frame



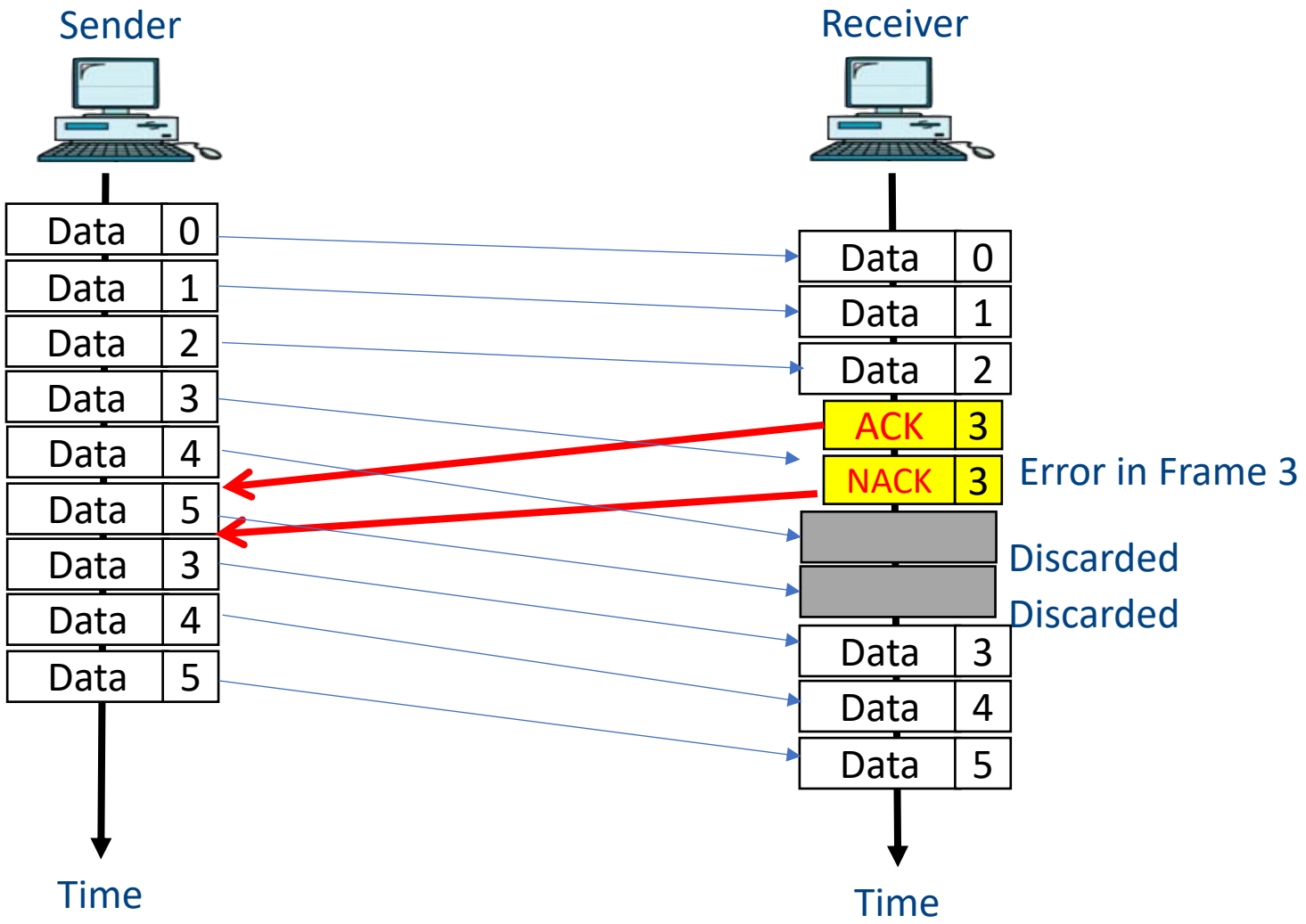
Stop-and-Wait ARQ Lost ACK



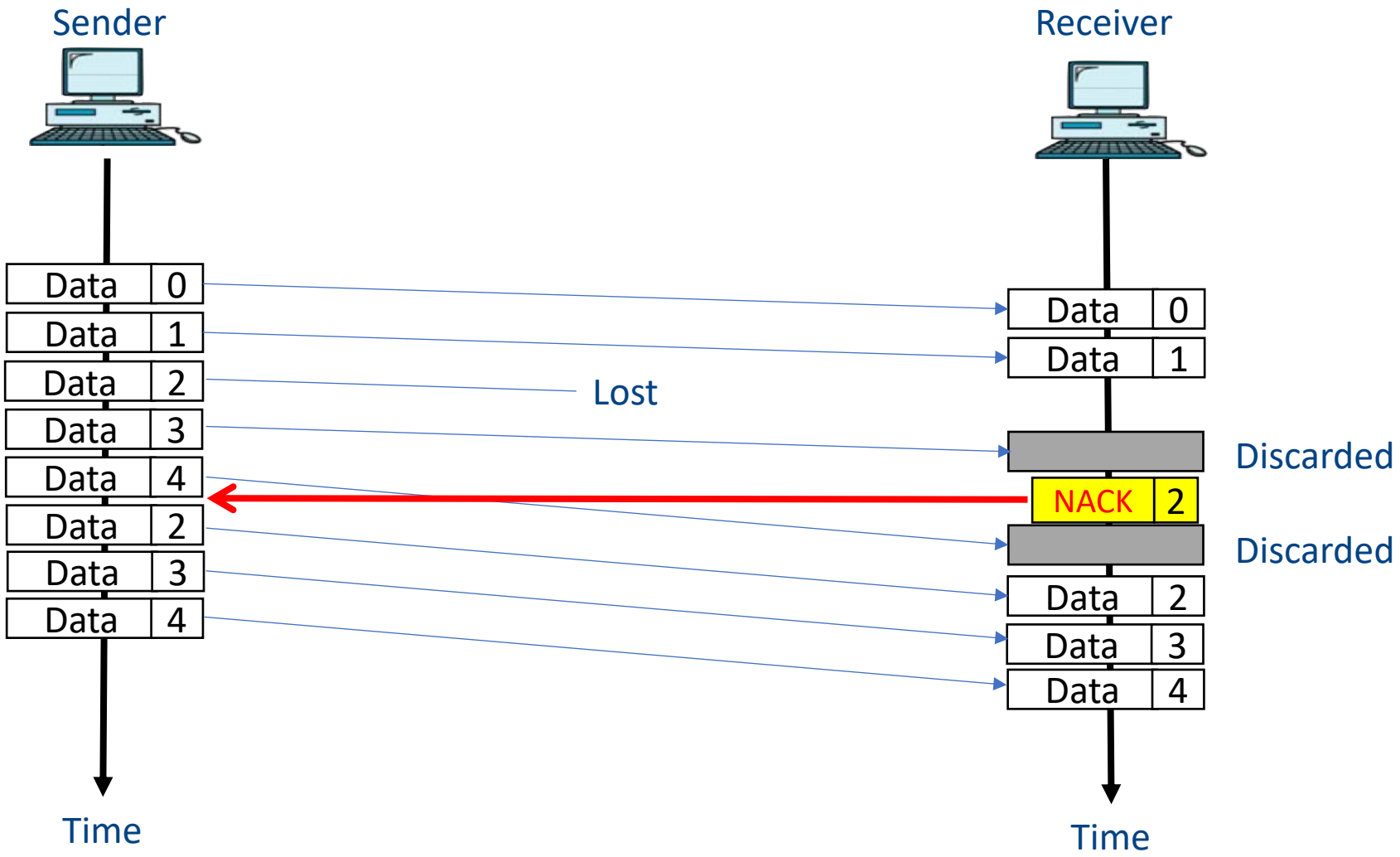
Go-Back-N ARQ Scheme

- Transmit frames continuously if possible
- N outstanding frames at most on the link

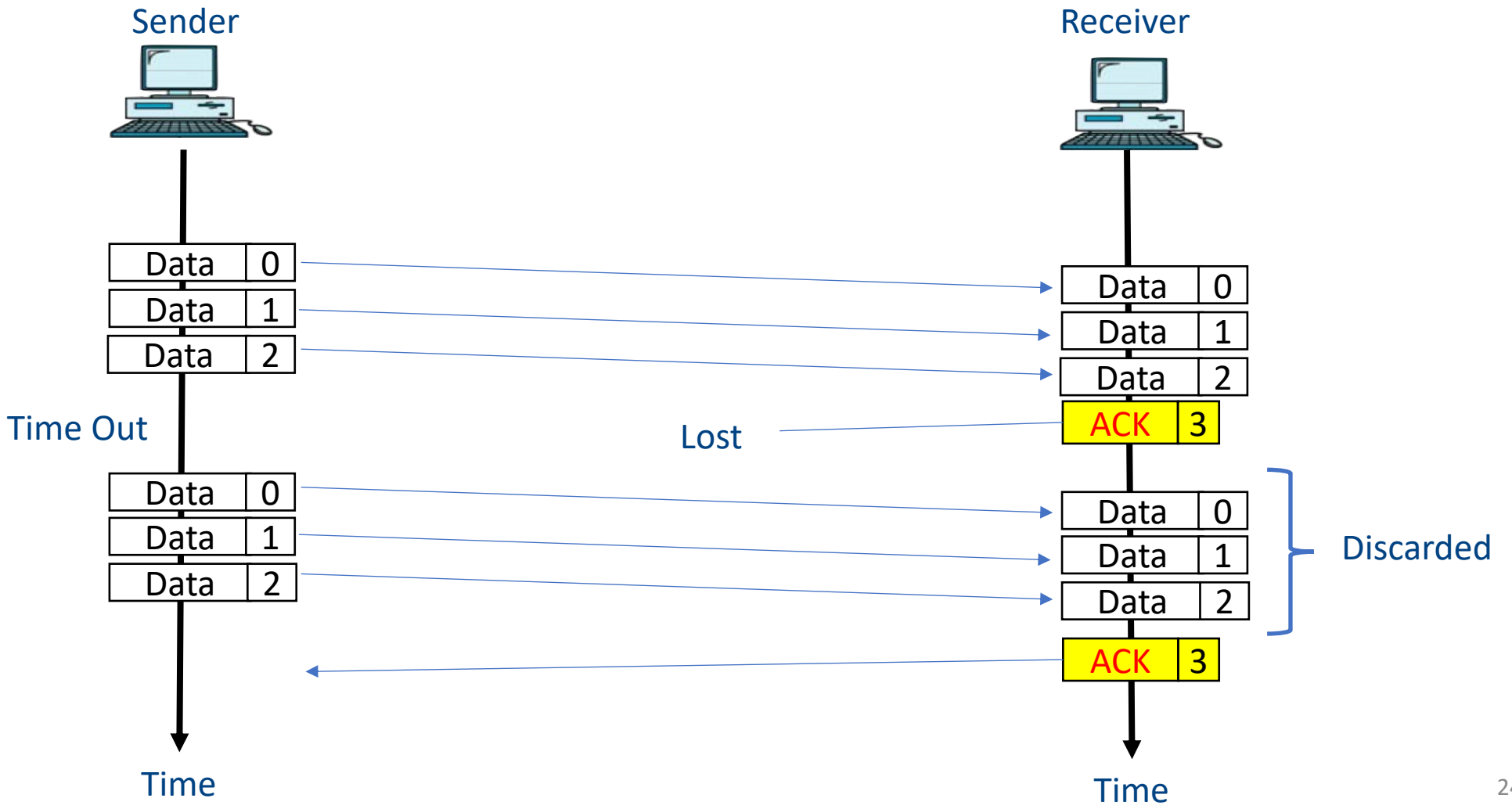
Go-Back-N ARQ Damaged Frame



Go-Back-N ARQ Lost Frame



Go-Back-N ARQ Lost ACK

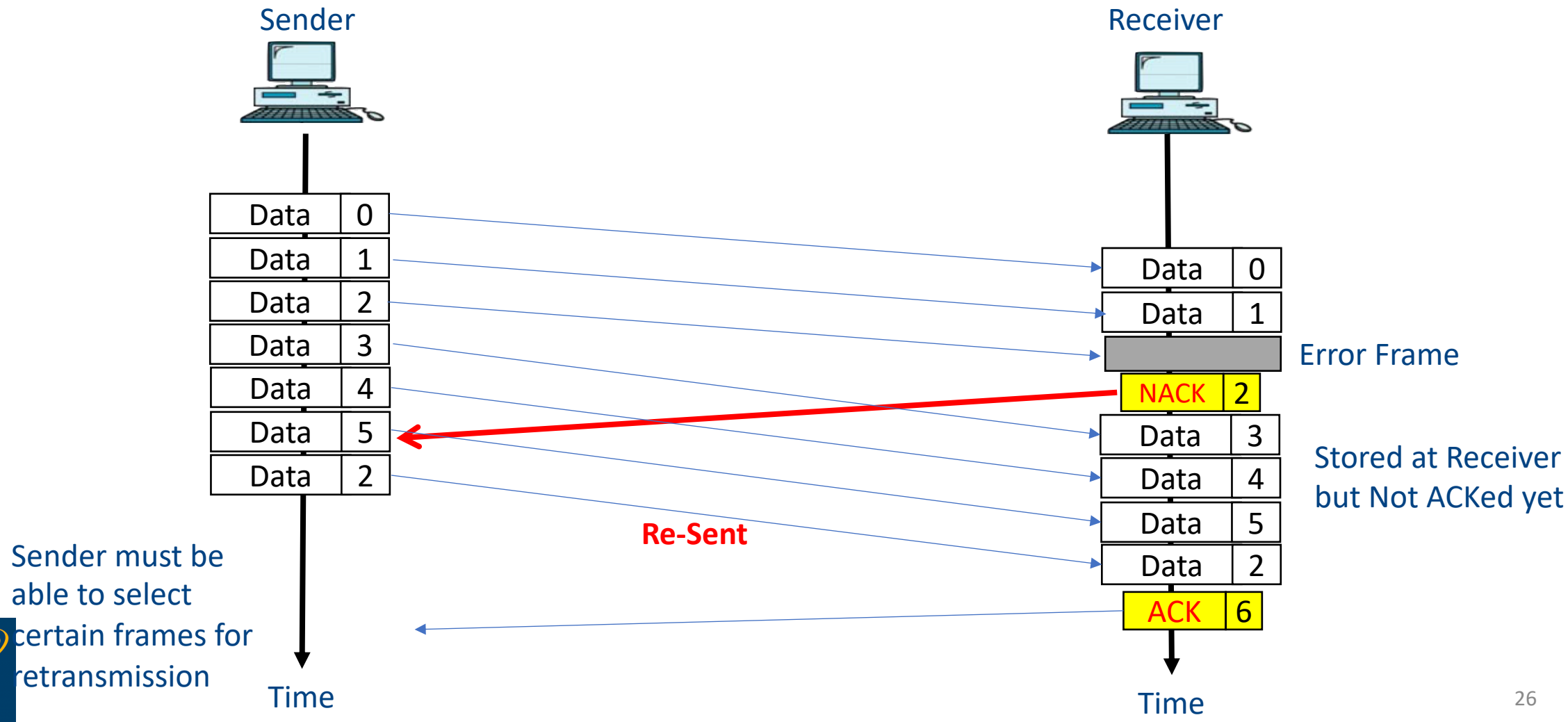


Go-Back-N ARQ Scheme

- Receiver only accepts correctly-received Frames in the correct order
  - so Receiver doesn't have to buffer any Frames and re-order them...
- If a Frame or an ACK is lost, all received frames after it will be discarded and transmitted again
- Why go back and retransmit all the frames?
  - Some might be good
  - Only retransmit the bad frames!
  - This is what we call **Selective Repeat**

DCU
Ollscoil Chathair Bhaile Átha Cliath
Dublin City University

Selective Repeat ARQ Damaged Frame



Selective Repeat

✓ Improvement over Go Back N

✓ Attains the theoretical maximum throughput

X Out of order Frames, so the Receiver must reorder them

X More complex Sender and Receiver, so expensive