

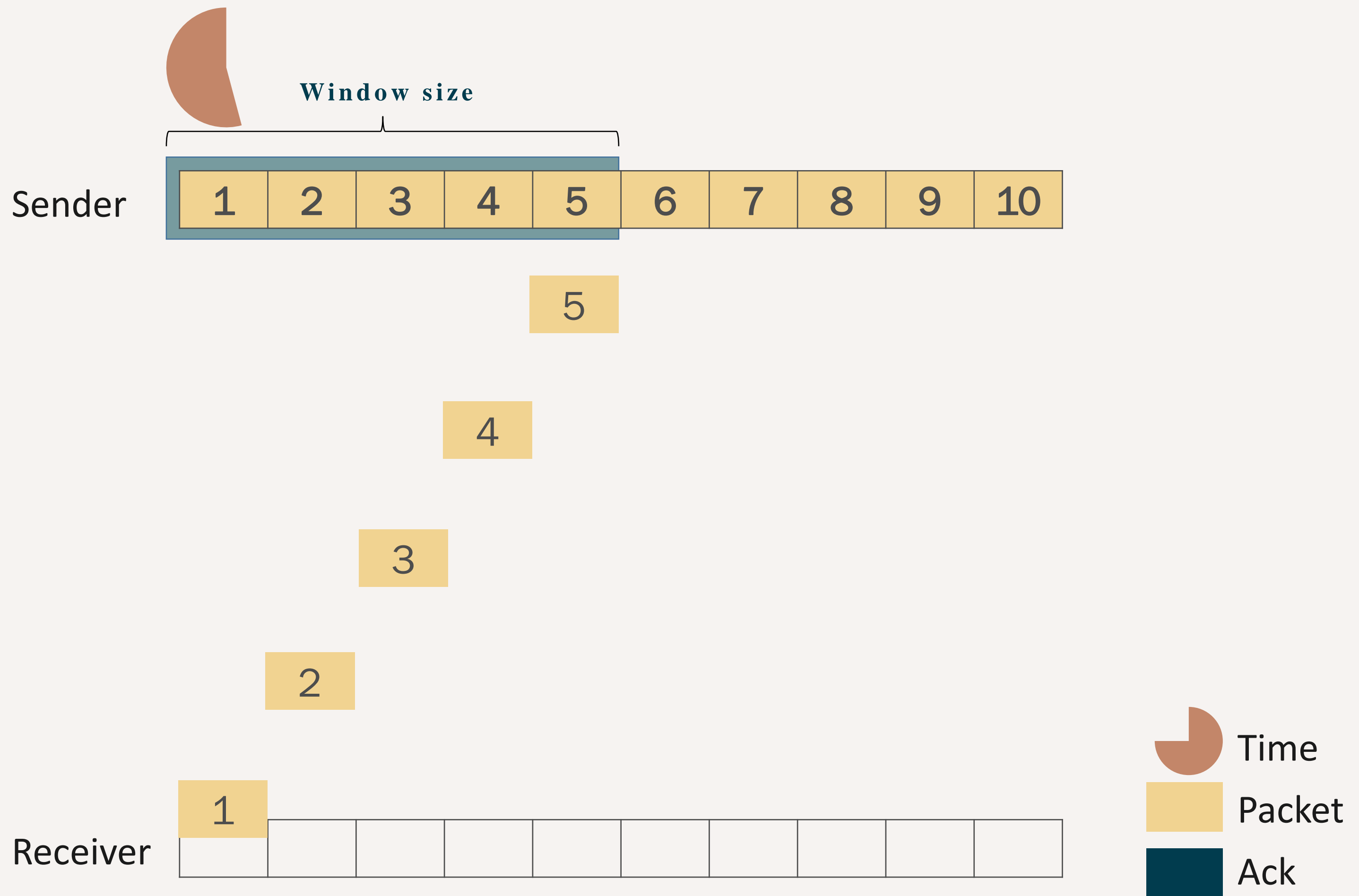
Ai-Chun Pang / Instructor

林偉哲 陳令原 / T.A.s

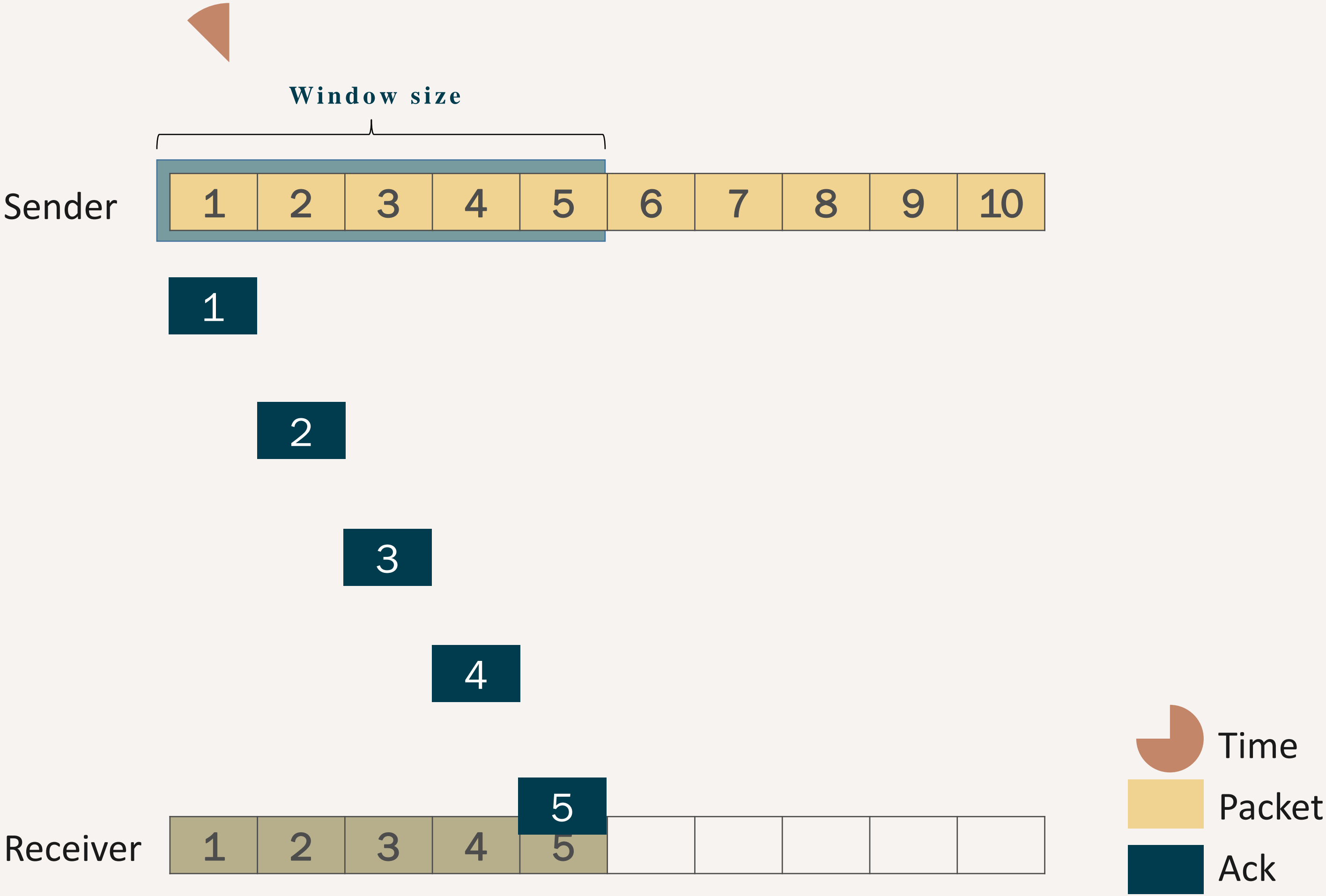
Assignment 3 - Retransmission & Congestion Control

What is Go-Back-N?

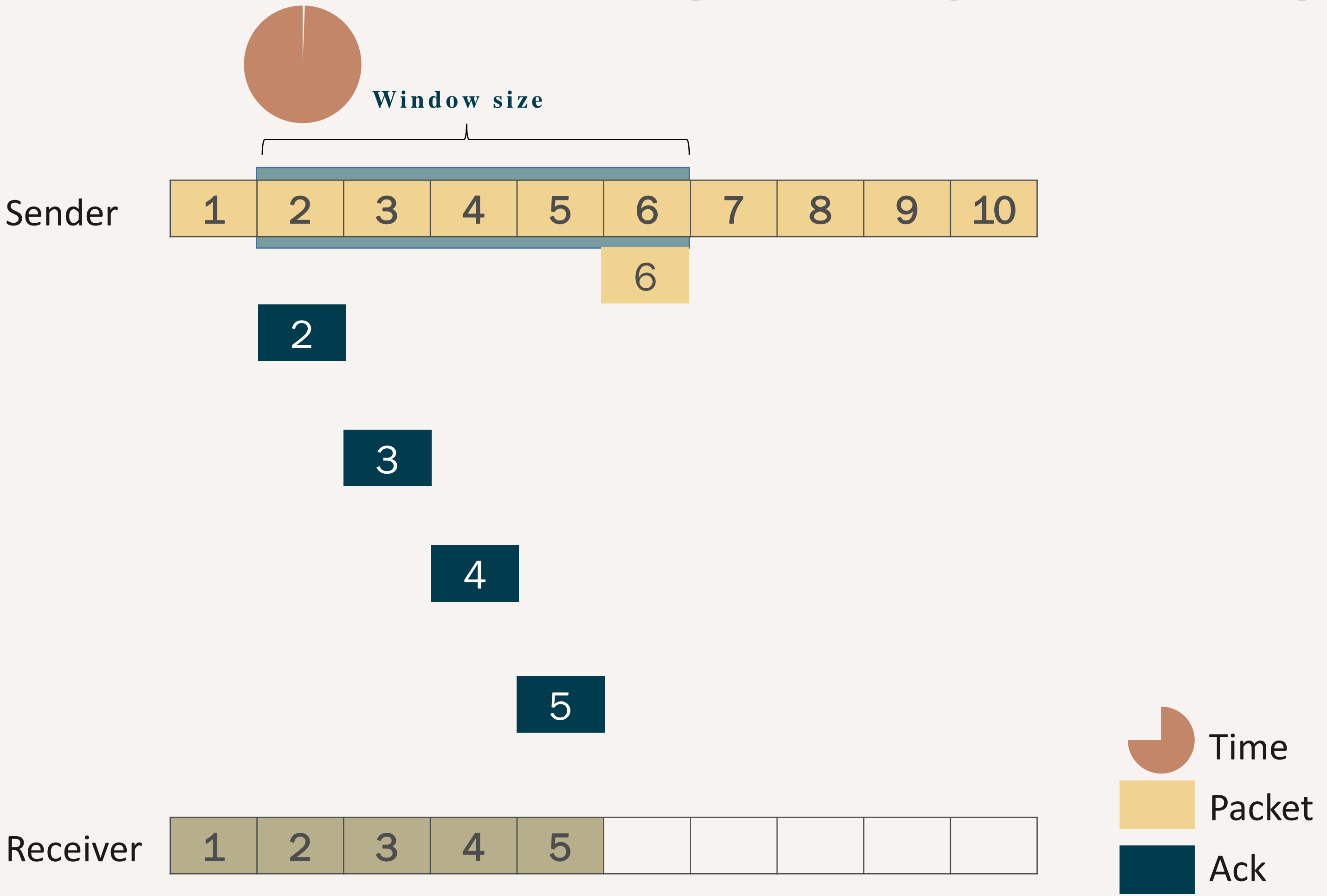
Go-Back-N case 1 (working normally)



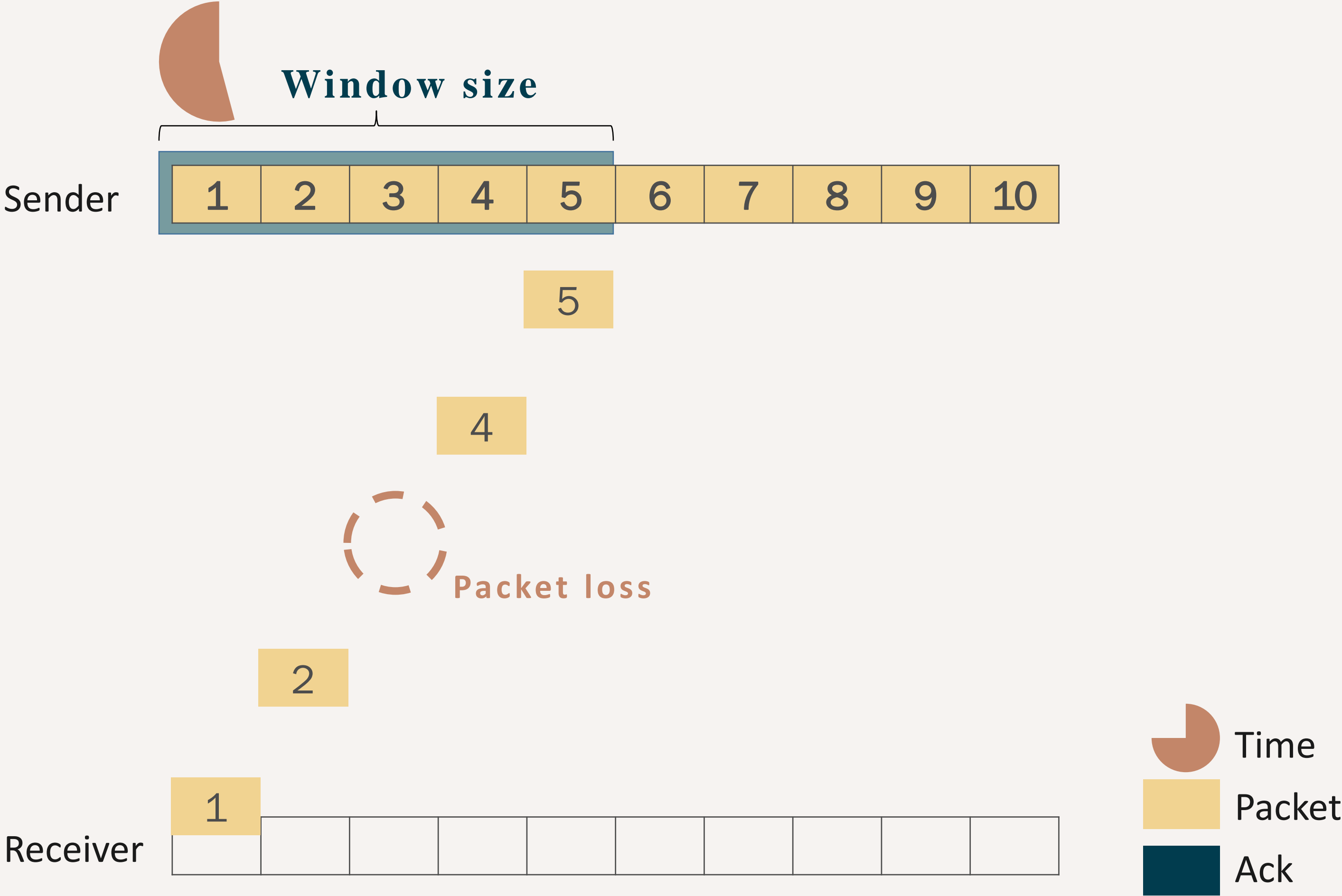
Go-Back-N case I (working normally)



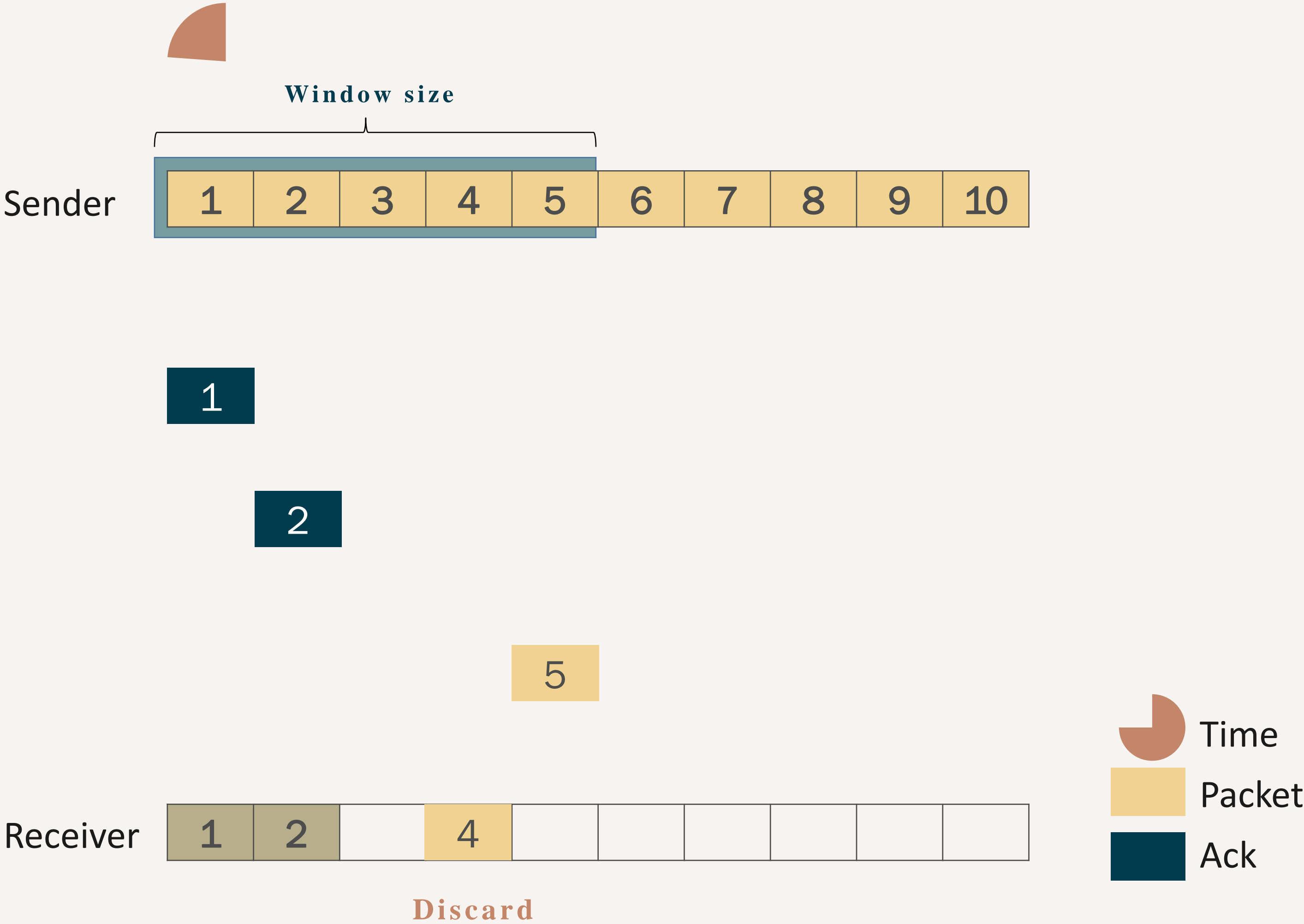
Go-Back-N case I (working normally)



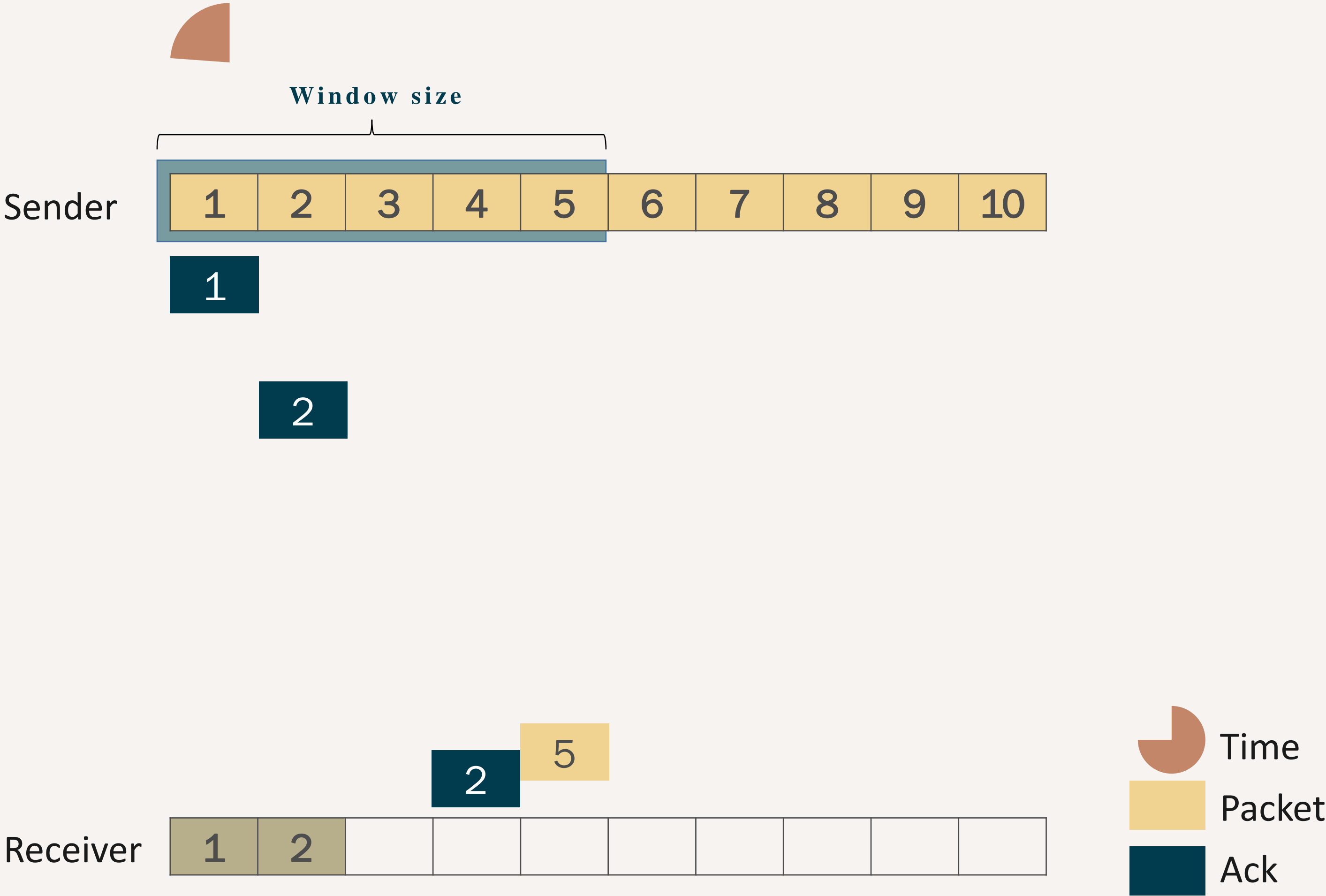
Go-Back-N case 2 (packet loss)



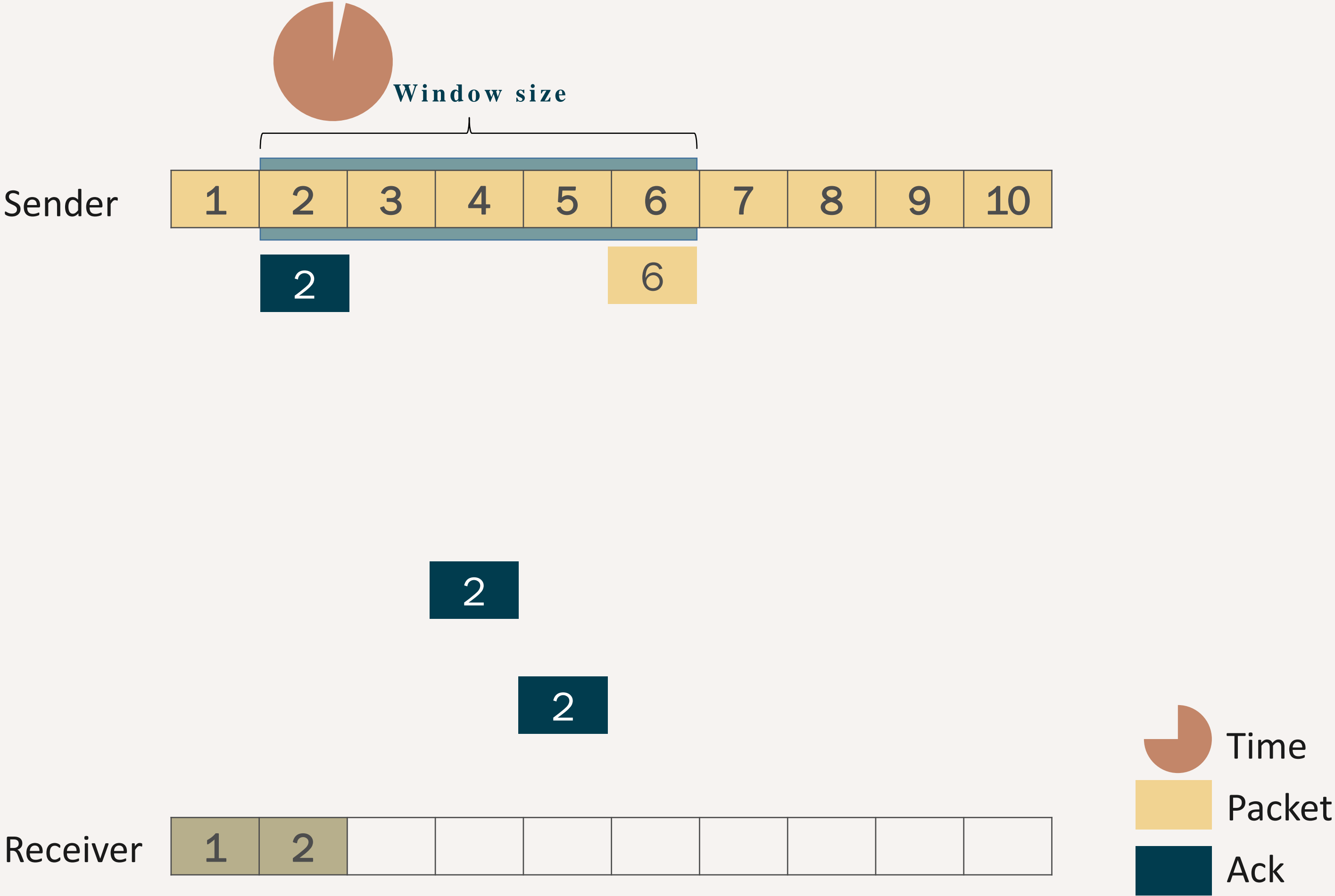
Go-Back-N case 2 (packet loss)



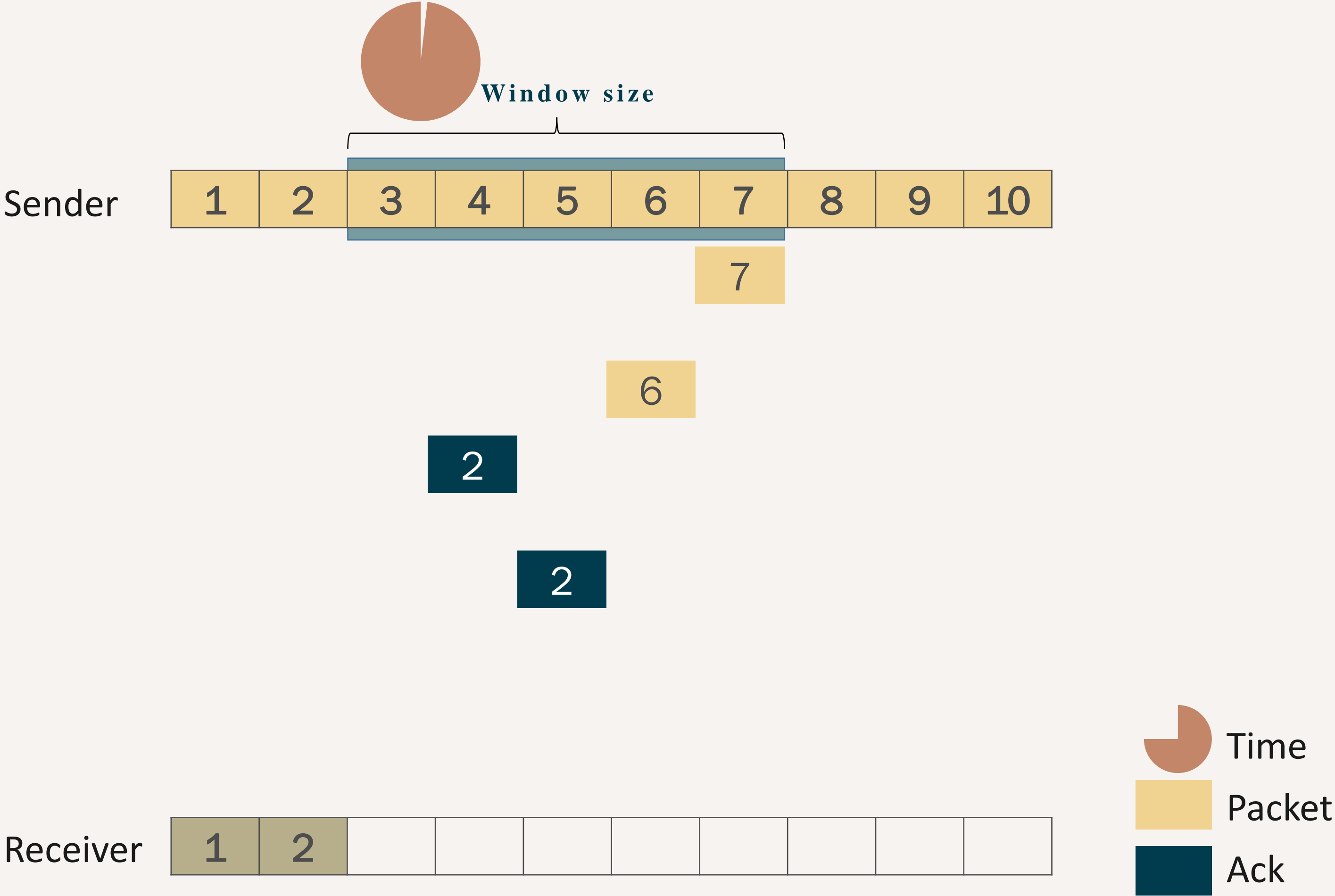
Go-Back-N case 2 (packet loss)



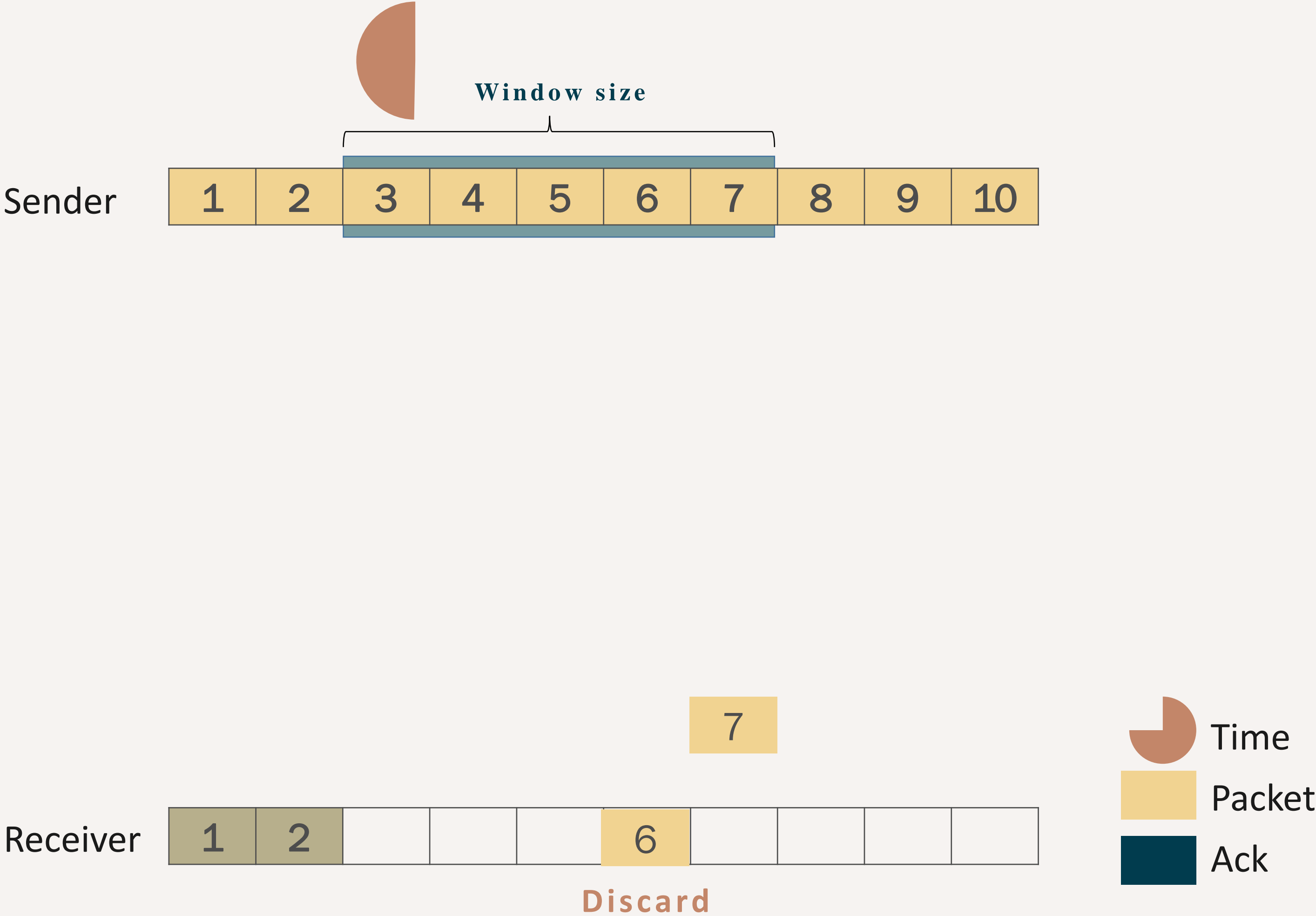
Go-Back-N case 2 (packet loss)



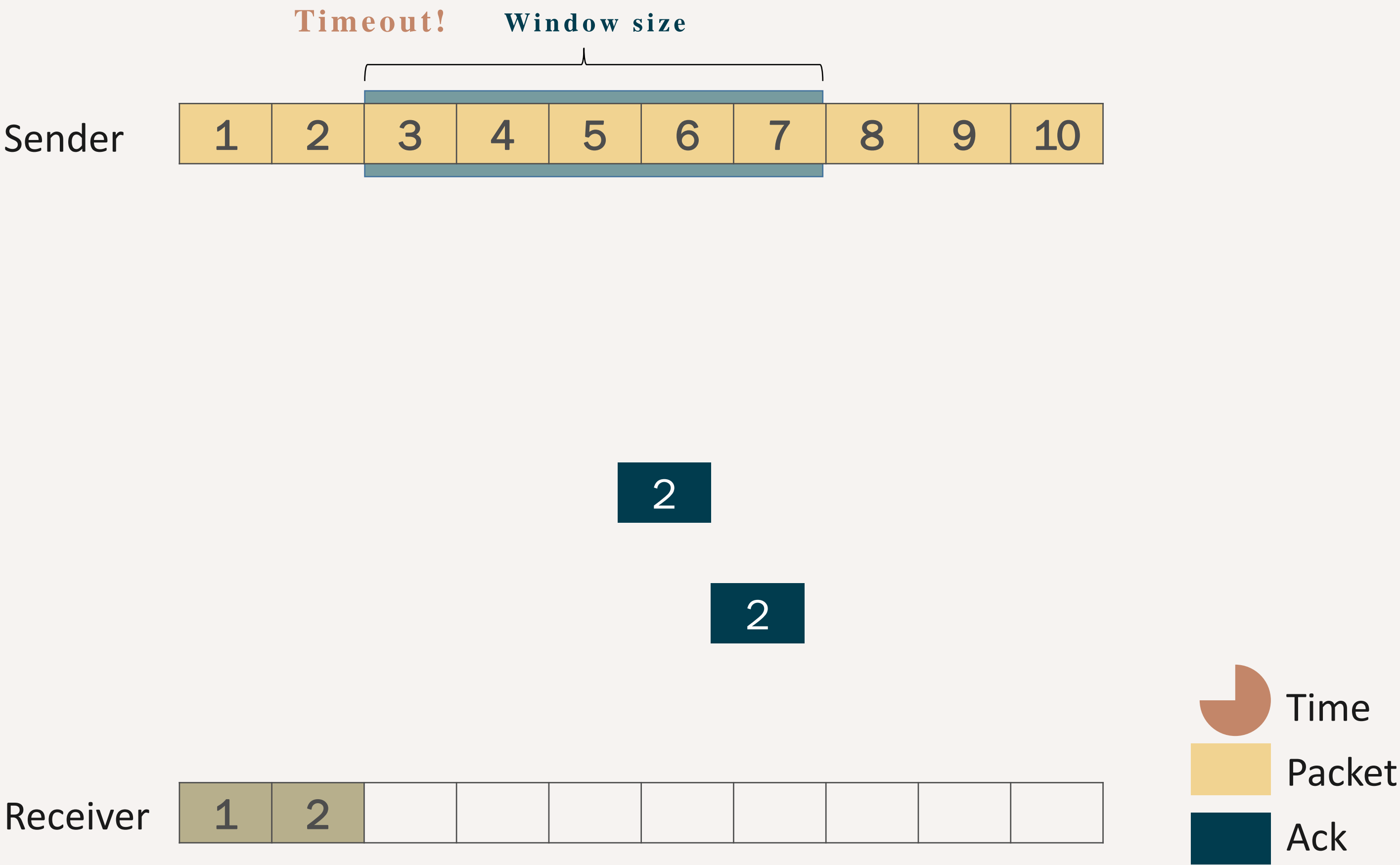
Go-Back-N case 2 (packet loss)



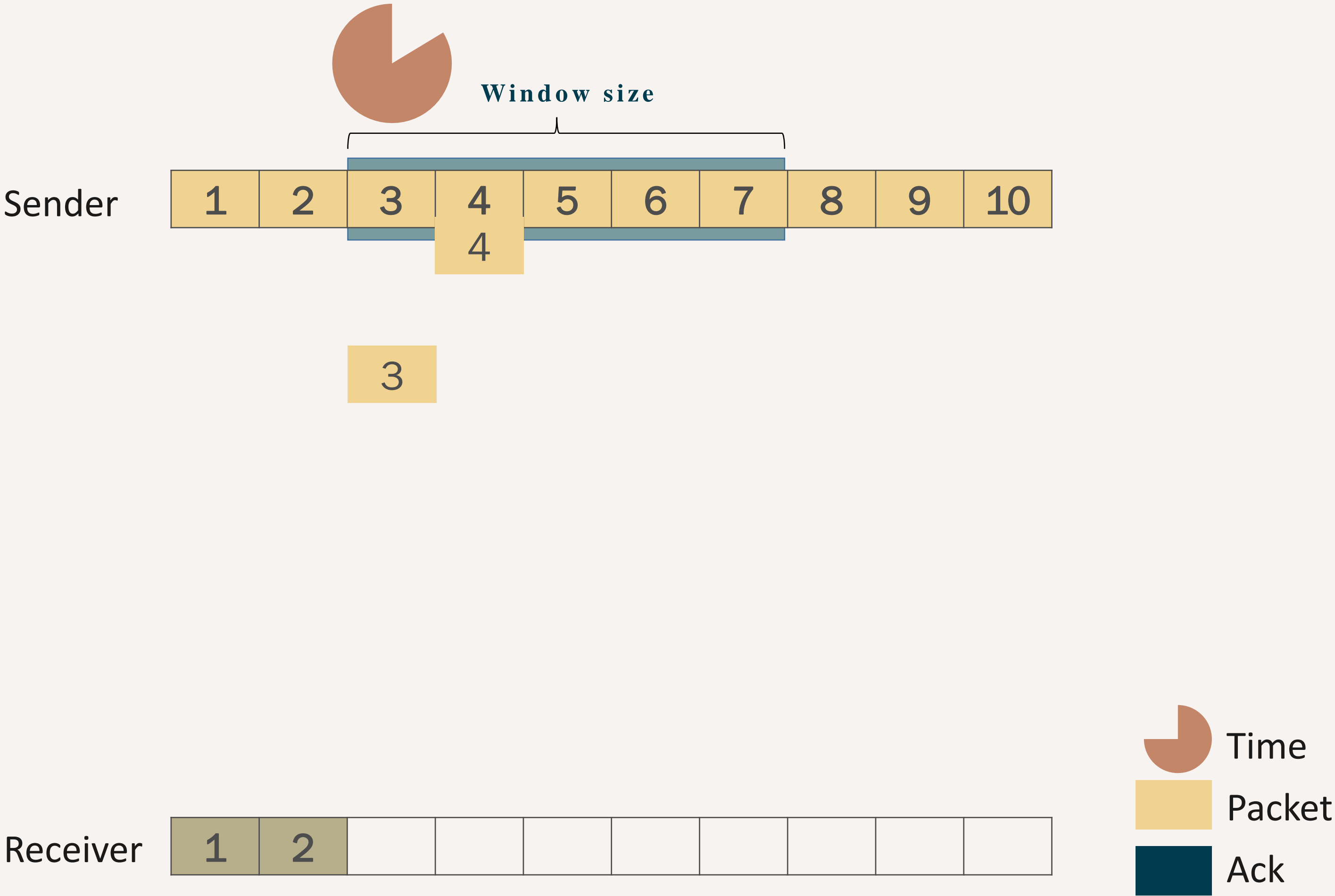
Go-Back-N case 2 (packet loss)



Go-Back-N case 2 (packet loss)



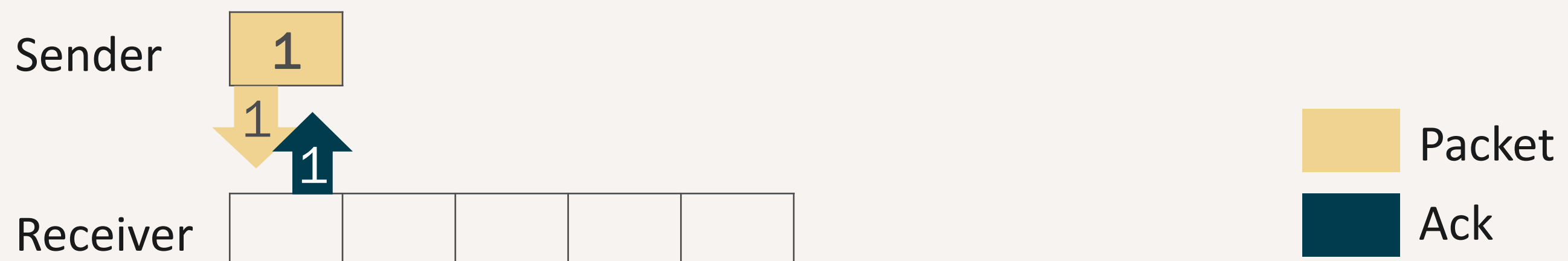
Go-Back-N case 2 (packet loss)



Go-Back-N with Congestion Control

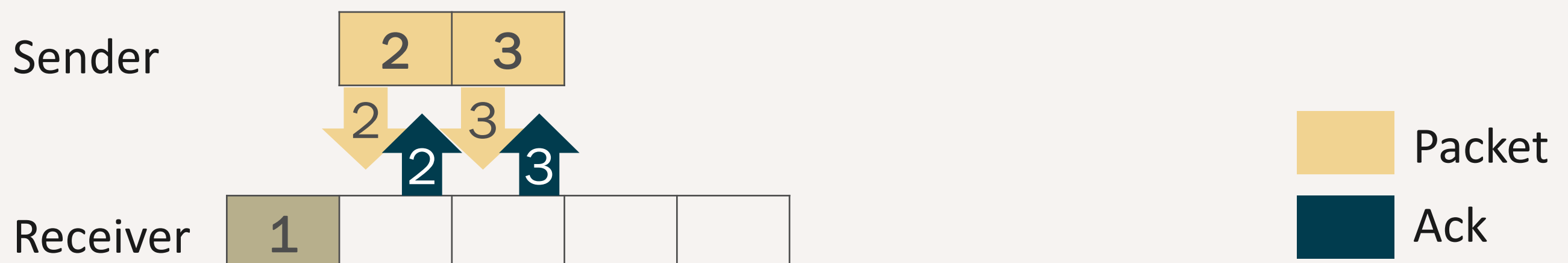
Go-Back-N + Congestion Control

- Sender sends Data 1
- Congestion window = 1. Threshold = 2
- Receiver sends ACK 1



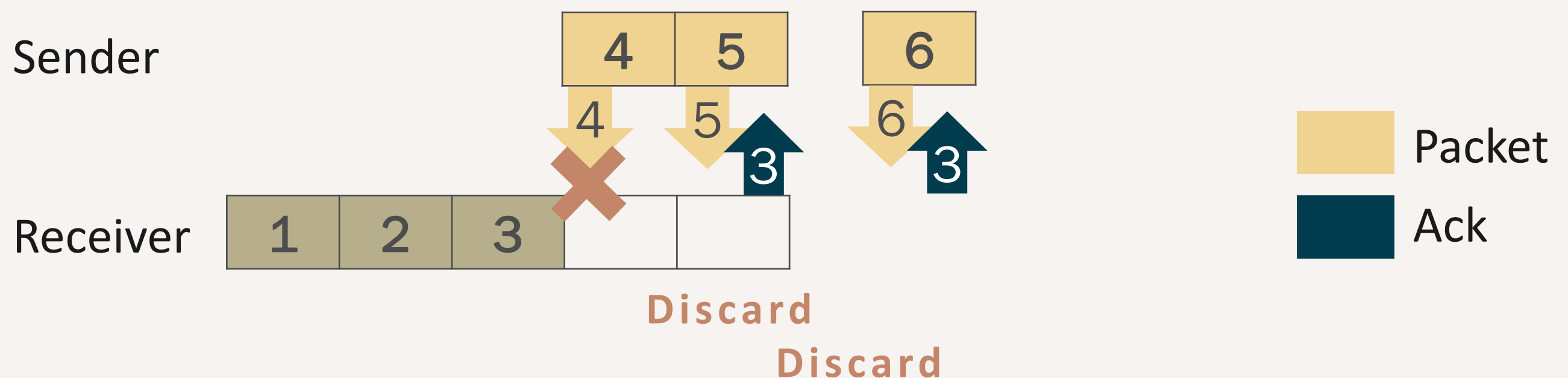
Go-Back-N + Congestion Control

- Sender sends Data 2,3
- Congestion window =2, Threshold =2;
- Receiver sends ACK 2,3



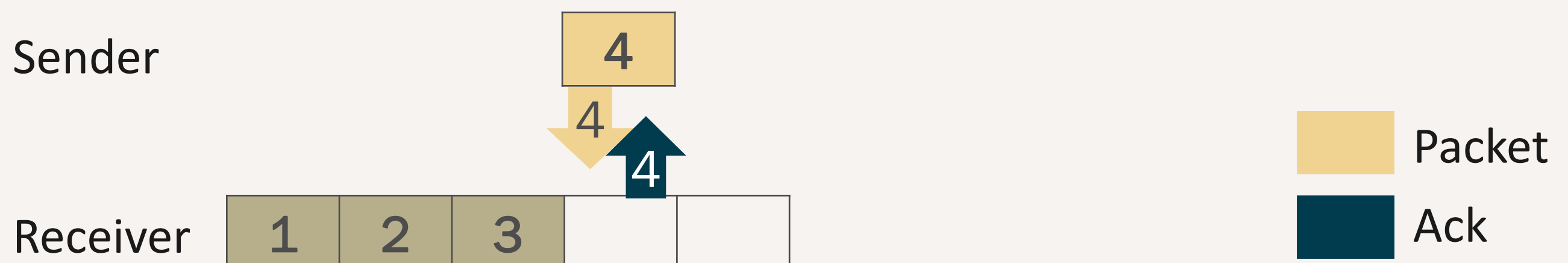
Go-Back-N + Congestion Control

- Sender sends Data 4,5,6
- Congestion window = 3; Threshold = 2;
- Receiver drops Data 5, sends ACK 3, drops Data 6, sends ACK 3



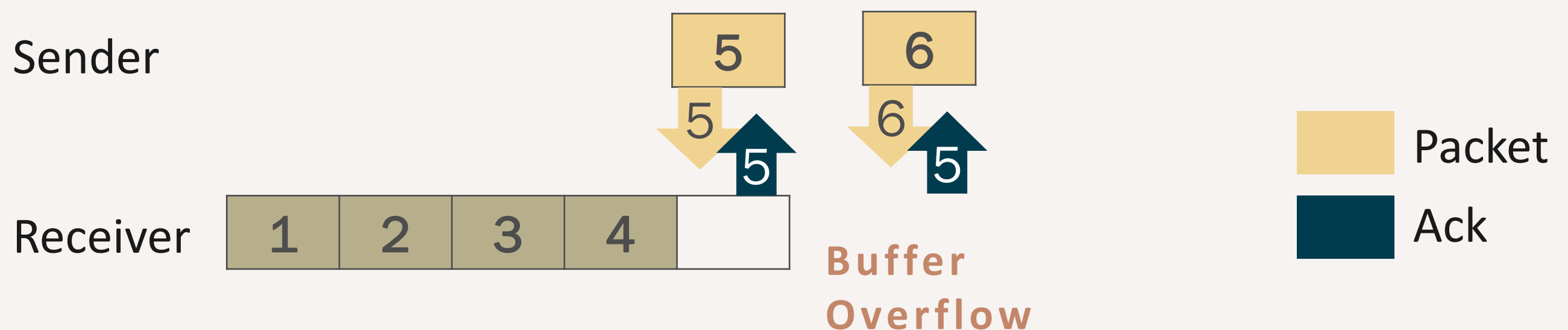
Go-Back-N + Congestion Control

- Sender sends Data 5,6
- Congestion window = 2, Threshold =1;
- Receiver sends ACK 5, drops Data 6, sends ACK 5, flush buffer ()



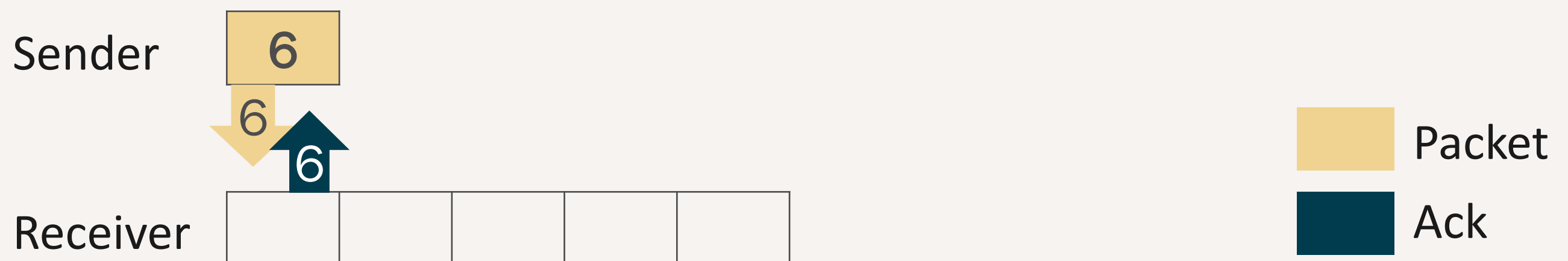
Go-Back-N + Congestion Control

- Sender sends Data 4,5,6
- Congestion window = 3; Threshold = 2;
- Receiver drops Data 5, sends ACK 3, drops Data 6, sends ACK 3



Go-Back-N + Congestion Control

- Sender sends Data 1
- Congestion window = 1. Threshold = 2
- Receiver sends ACK 1

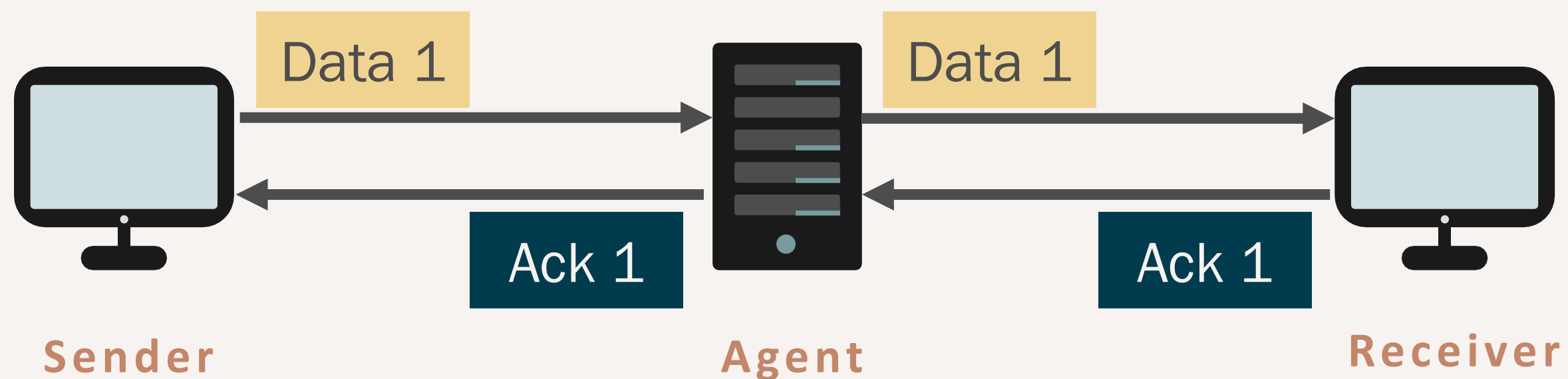


Assignment 3

Announcement

Specification (1/9)

- Implement three components in **C/C++** with **UDP socket**: sender, receiver and agent.



- Sender / Receiver**
 - Send / receive **video frame** by UDP
 - Provide reliable transmission
 - Congestion control
- Agent**
 - Forward Data & ACK packets
 - **Randomly drop data packet**
 - Compute loss rate

Specification (2/9)

- **Reliable Transmission**
 - Data & ACK
 - Time out & Retransmission(**Go-Back-N**)
 - **Sequence number**
 - Completeness of transmitted file
- **Buffer handling [**receiver side**]**
 - Buffer Overflow:
 - Drop the packet** in case of **out of buffer**
 - Flush (write) to the file:
 - Only when **both buffer overflows** and **all packets in range** are received.

Specification (3/9)

- Congestion Control (sender sider)
 - Slow Start
 1. Send single packet in the beginning
 2. When window size is under the threshold, it increases **exponentially** until packet loses
 3. When window size is over the threshold, it increases **linearly** until packet loses
 - Packet loss / Time out
 1. Set **threshold** to $\max\left(\left\lceil \frac{\text{window size}}{2} \right\rceil, 1\right)$
 2. Set **window size** to 1
 3. Retransmit – from the first “unACKed packet”

Specification (4/9)

- **Show Message**

- **Sender:**
send, recv, data, ack, fin, finack, sequence number, time out, resnd, winSize, threshold
- **Receiver:**
send, recv, data, ack, fin, finack, sequence number, drop, flush
- **Agent:**
get, fwd, data, ack, fin, finack, sequence number, drop, loss rate

Specification (5/9)

- Show Message
 - Sender:

```
send    data    #1,    winSize = 1
recv    ack     #1
send    data    #2,    winSize = 2
send    data    #3,    winSize = 2
recv    ack     #2
recv    ack     #3
send    data    #4,    winSize = 3
send    data    #5,    winSize = 3
send    data    #6,    winSize = 3
recv    ack     #3
recv    ack     #3
time    out,    threshold = 1
resnd   data    #4,    winSize = 1
recv    ack     #4
resnd   data    #5,    winSize = 2
resnd   data    #6,    winSize = 2
recv    ack     #5
recv    ack     #5
time    out,    threshold = 1
resnd   data    #6,    winSize = 1
recv    ack     #6
send    fin
recv    finack
```

Specification (6/9)

- Show Message

- Agent:

```
get    data    #1
fwd    data    #1,    loss rate = 0.0000
get    ack     #1
fwd    ack     #1
get    data    #2
fwd    data    #2,    loss rate = 0.0000
get    data    #3
fwd    data    #3,    loss rate = 0.0000
get    ack     #2
fwd    ack     #2
get    ack     #3
fwd    ack     #3
get    data    #4
drop   data    #4,    loss rate = 0.2500
get    data    #5
fwd    data    #5,    loss rate = 0.2000
get    data    #6
fwd    data    #6,    loss rate = 0.1667
get    ack     #3
fwd    ack     #3
get    ack     #3
fwd    ack     #3
get    data    #4
fwd    data    #4,    loss rate = 0.1429
get    ack     #4
fwd    ack     #4
get    data    #5
fwd    data    #5,    loss rate = 0.1250
get    data    #6
fwd    data    #6,    loss rate = 0.1111
get    ack     #5
fwd    ack     #5
get    ack     #5
fwd    ack     #5
get    data    #6
fwd    data    #6,    loss rate = 0.1000
get    ack     #6
fwd    ack     #6
get    fin
fwd    fin
get    finack
fwd    finack
```

Specification (7/9)

- Show Message
 - Receiver:

```
recv    data    #1
send    ack     #1
recv    data    #2
send    ack     #2
recv    data    #3
send    ack     #3
drop    data    #5
send    ack     #3
drop    data    #6
send    ack     #3
recv    data    #4
send    ack     #4
recv    data    #5
send    ack     #5
drop    data    #6
send    ack     #5
flush
recv    data    #6
send    ack     #6
recv    fin
send    finack
flush
```


Specification (8/9)

- Show Message

- The format used for transmission should be the same as follow:

fin: 0 or 1

syn: 0 or 1 (just make it 0)

ack: 0 or 1

```
21 typedef struct{
22     int length;
23     int seqNumber;
24     int ackNumber;
25     int fin;
26     int syn;
27     int ack;
28 } header;
29
30 typedef struct{
31     header head;
32     char data[1000];
33 } segment;
```

Specification (9/9)

- **Settings**

- Sender
 - Arguments: IP, Port, path of source file,... etc.
 - Default threshold:16
- Receiver
 - Arguments: IP, port, ... etc.
 - Default buffer size: 32 segments
- Agent
 - Arguments: IP, port, loss rate, ... etc.
- Data packet size (payload): 1KB
- Time out:
Less than or equal to 1 sec ($\leq 1 \text{ sec}$)

Grading Policy (1/2)

- This assignment accounts for 13% of the total score.
- Video Streaming (10%)
 - Correctly play the example video of HW2 (5%)
 - Correctly play a resolution-unknown video (5%)
- Reliable transmission (20%)
- Congestion control (25%)
- Buffer handling (15%)
- Agent (9%)
 - Randomly drop data packet (5%)
 - Compute loss rate (4%)
- Show Message (9%)
 - Show message correctly (3%*3)
- Report (12%)
 - How to execute your program (3%)
 - Explain your program structure (3%*3)
(including **3 flow charts** for **sender**, **agent**, and **receiver**)

Grading Policy (2/2)

- **Submission**

- Your report format must be in **“.pdf”** format and named “report.pdf”, or else **you will get zero point** in the part.
- Please put all the files **into a folder** named **hw3_<student id>**, and compress the folder as a **.zip** file, and then submit the **.zip** file to **here**. The password is **<student id>** (alphabet is in uppercase).
- If we **cannot compile or execute your code**, you will **have a chance to demo your results** in your own environment.
- The penalty for **wrong format** is **10 points**.
- **No plagiarism** is allowed. A plagiarist will be graded **zero**.

- **Deadline**

- Due Date: **23:59:59, December 31st, 2019**
- Penalty for late submission after hard deadline is **“20 points per day”**