Computer network HW2

-Retransmission + Congest control

DATE : 2017/10/31

Introduction

Introduction

Target

- Application layer reliable transfer / congestion control
- Implement TCP by ***UDP***
- Socket Programming

UDP	TCP
Unreliable Unordered delivery	Reliable In-order Delivery Congestion Control

Introduction

You need to implement three components : the sender, receiver and agent.

Introduction

Sender / Receiver

- Send / receive file by UDP
- Provide reliable transmission
- Congestion control

Agent

- Forward Data & ACK packets
- Randomly drop data packet
- Compute loss rate

Requirement

Requirement

Reliable Transmission

- Data & ACK
- Time out & Retransmission(***Go-Back-N***)
- Sequence number
- Completeness and correctness of transmitted file

Requirement

Congestion control [***sender side***]

- Slow start
  - Send single packet in the beginning
  - When below the threshold, congestion window increase exponentially until packet lose, i.e.,1→2→4→8→…
  - When larger than or equal to the threshold, congestion window increase linearly until packet loss, i.e.,16→17 →18→19→…
- Packet loss / time out
  - Set threshold to max($\left\lfloor \frac{congestion\ window}{2} \right\rfloor$, 1)
  - Set Congestion window to 1
  - Retransmit
    - From the first “un-ACKED packet”

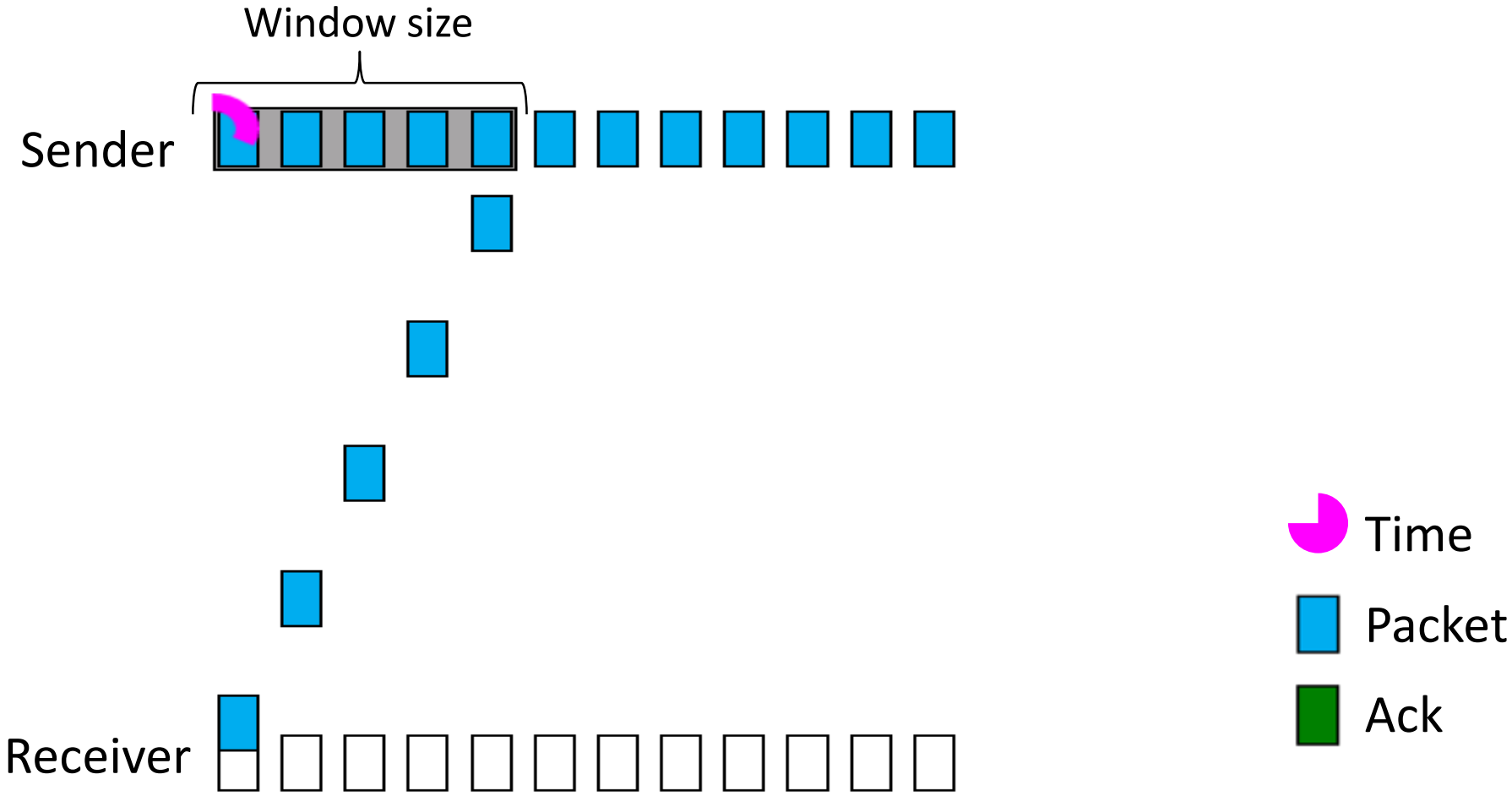
Requirement

Buffer handling [***receiver side***]

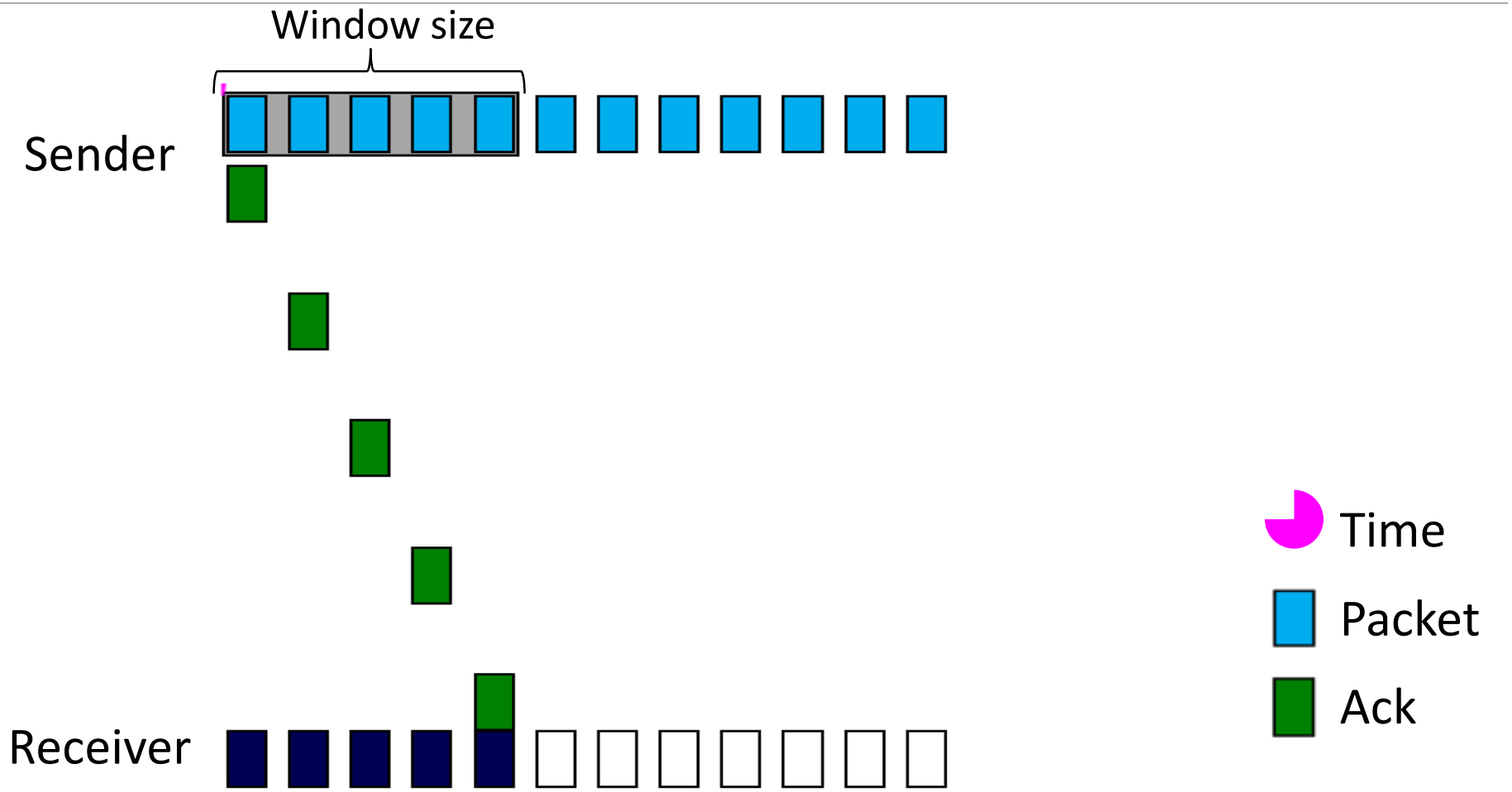
- Buffer Overflow
  - Drop packet if “out of range” of buffer
- Flush (write) to the file
  - Only when ***both buffer overflows and all packets in range are received***.

What is Go-Back-N(GBN)?

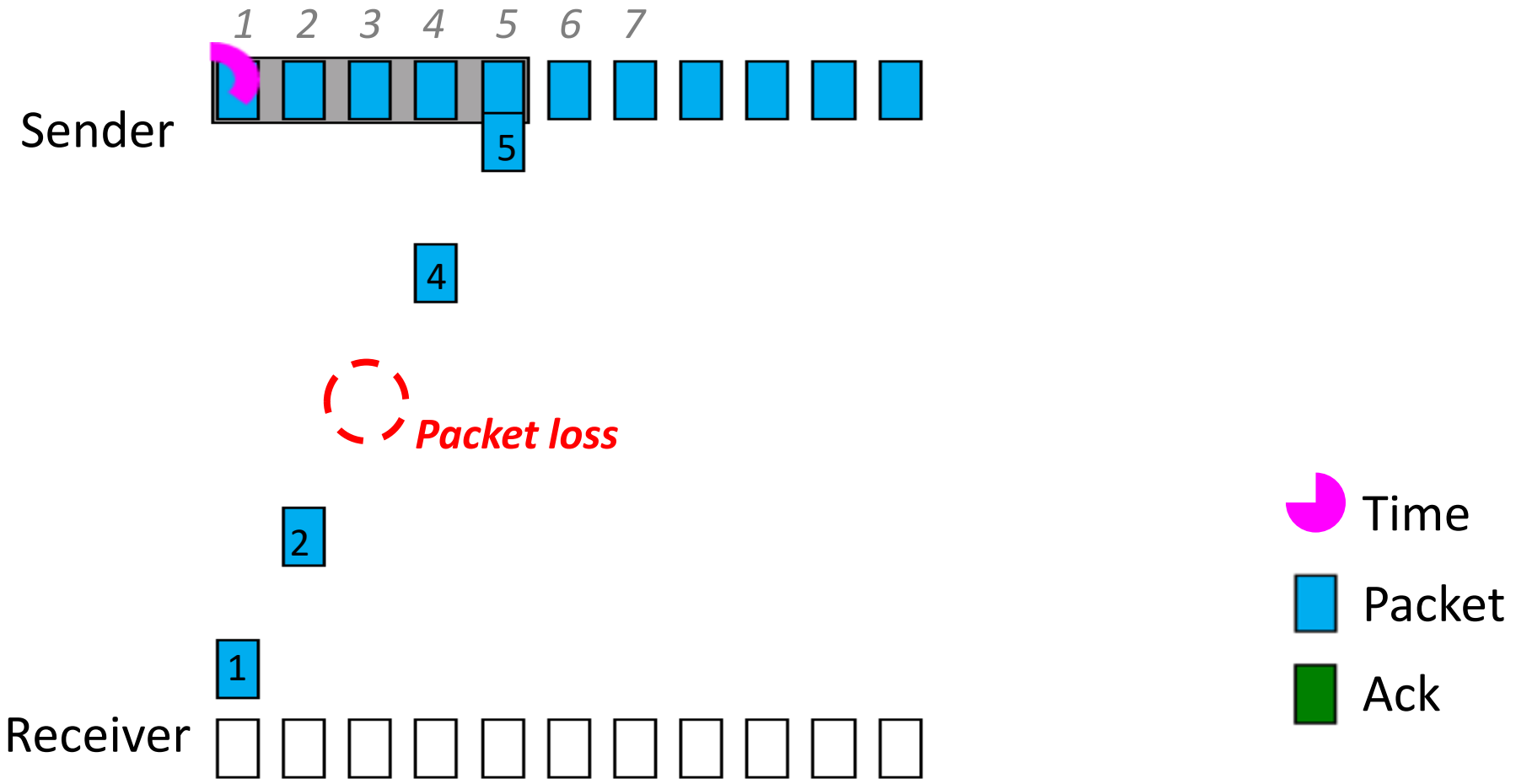
Go-Back-N case 1 (working normally)



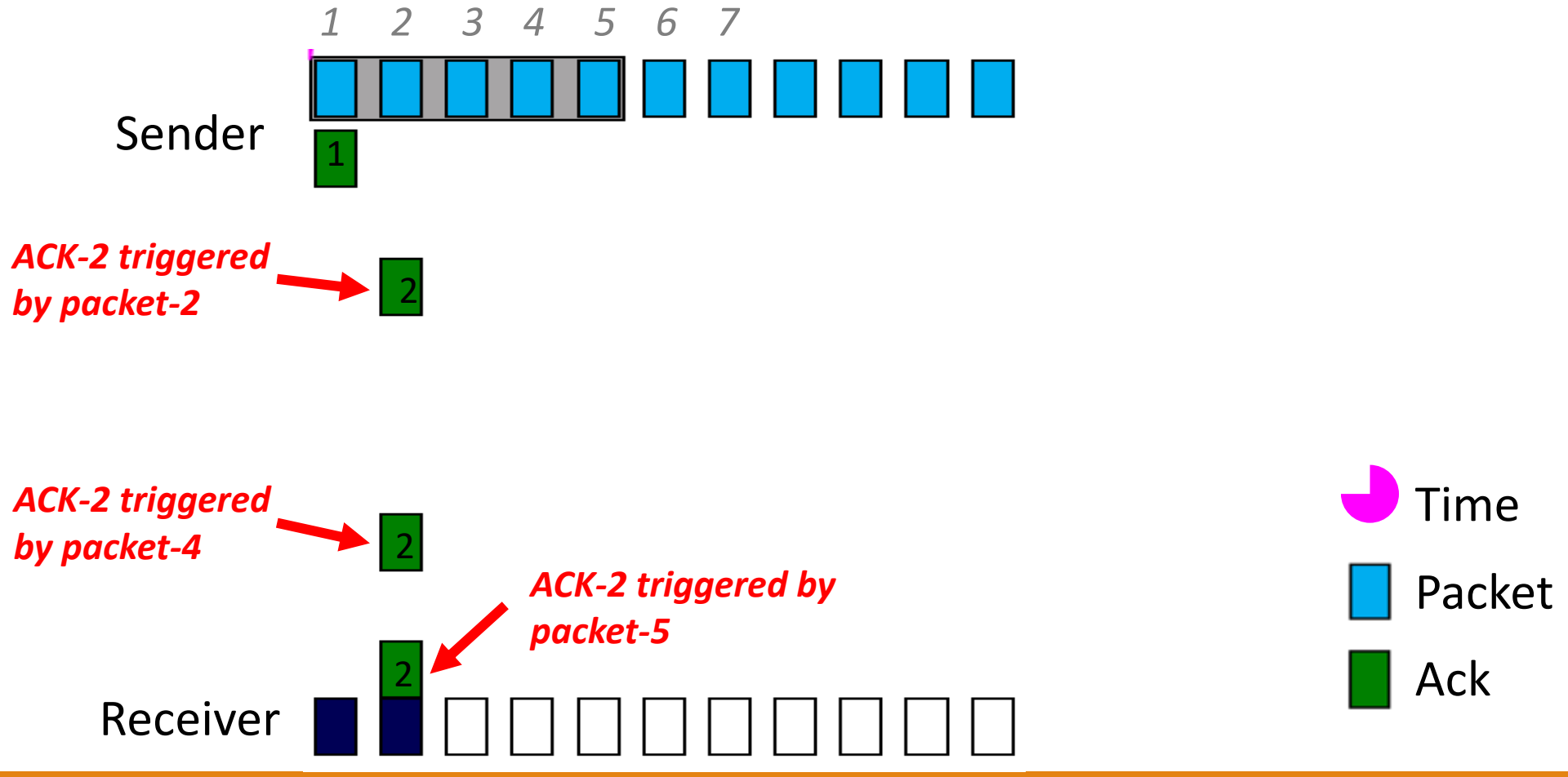
Go-Back-N case 1 (working normally)



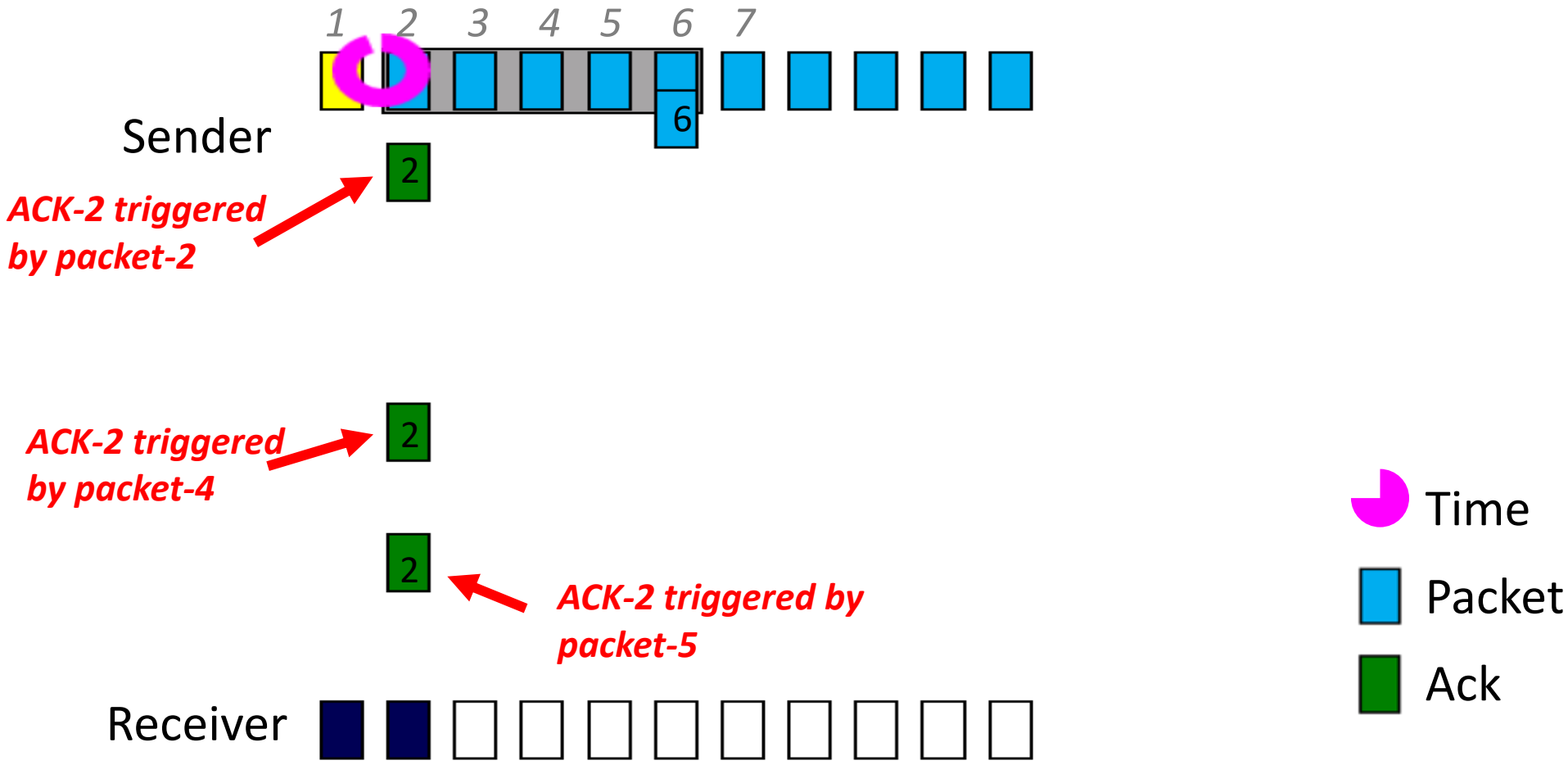
Go-Back-N case 2 (packet loss)



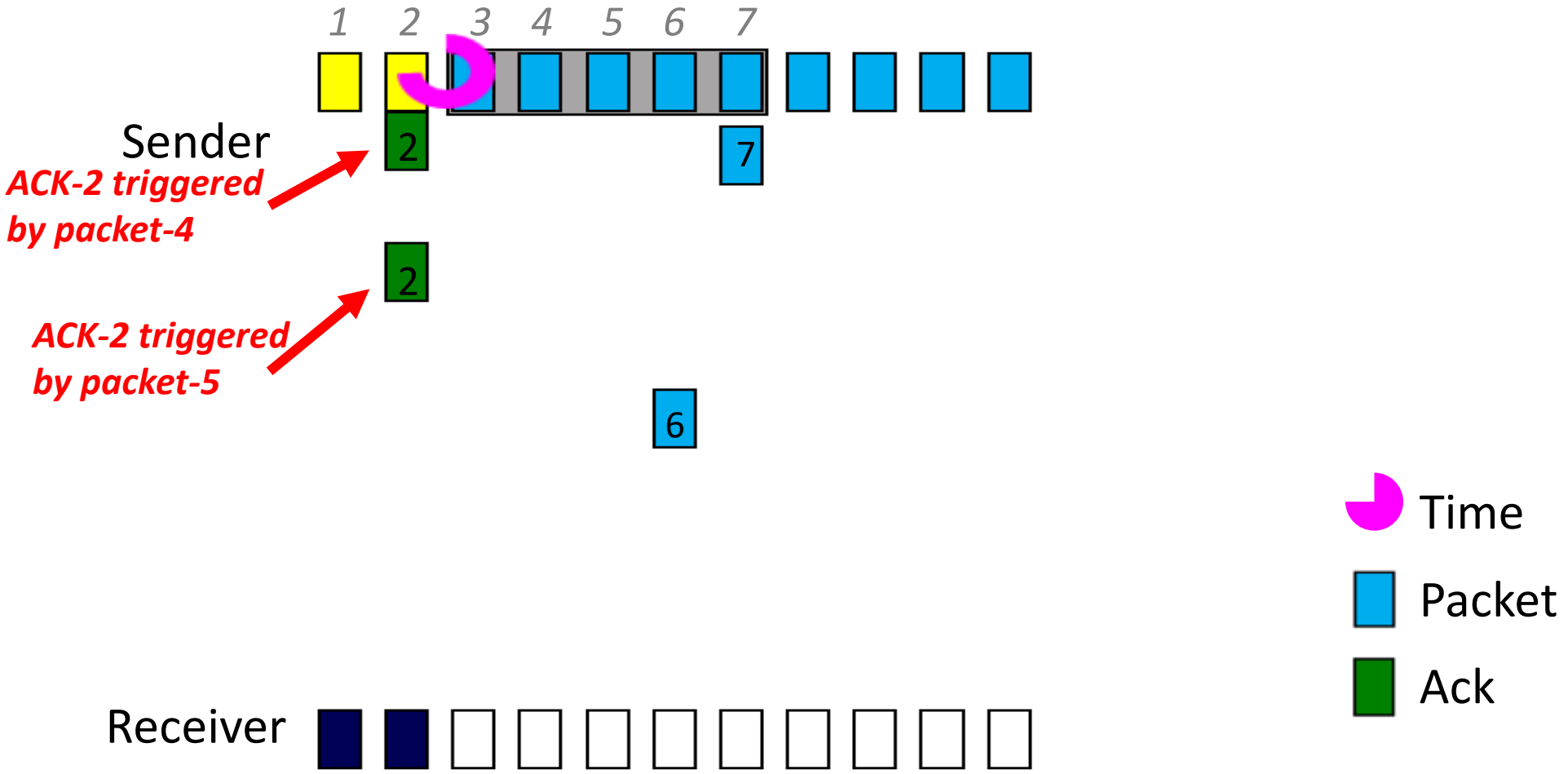
Go-Back-N case 2 (packet loss)



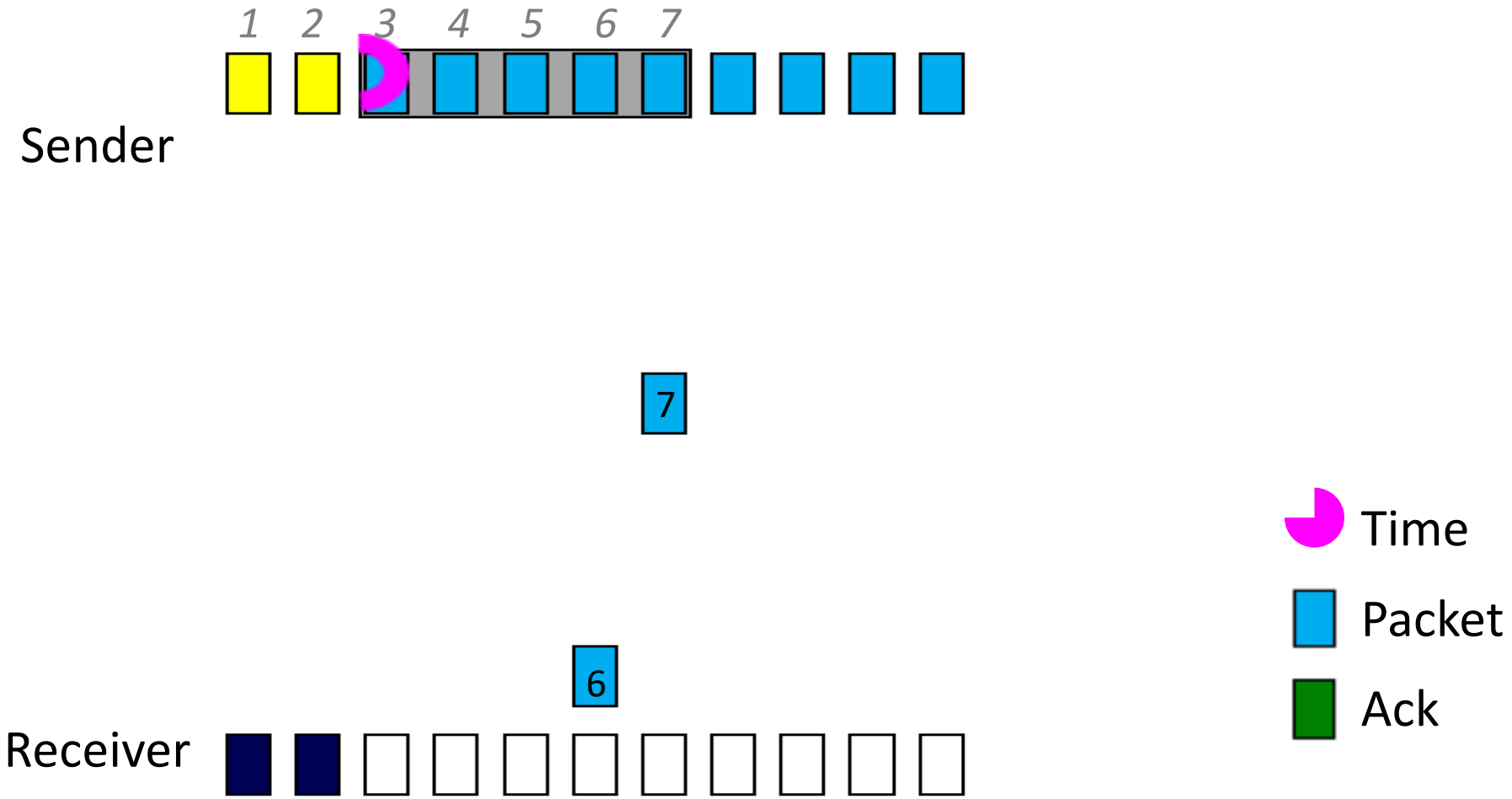
Go-Back-N case 2 (packet loss)



Go-Back-N case 2 (packet loss)

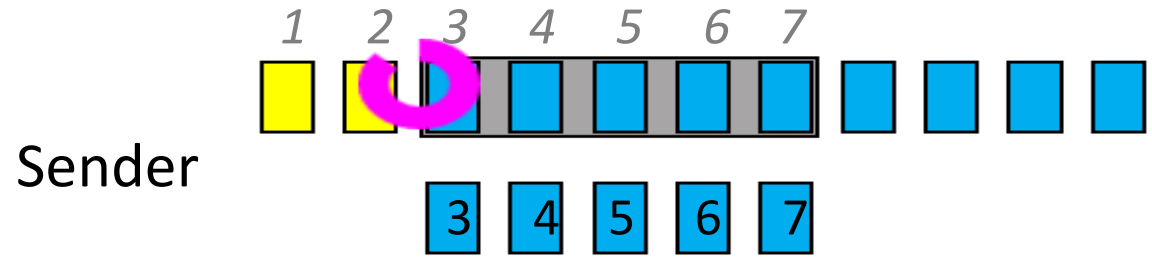


Go-Back-N case 2 (packet loss)



Go-Back-N case 2 (packet loss)

Go-Back-N case 2 (packet loss)



Receiver

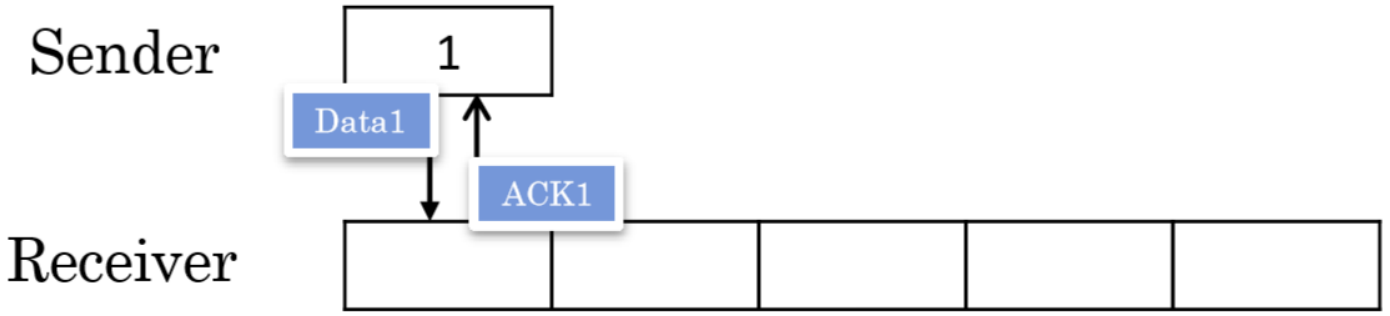
Time

Packet

Ack

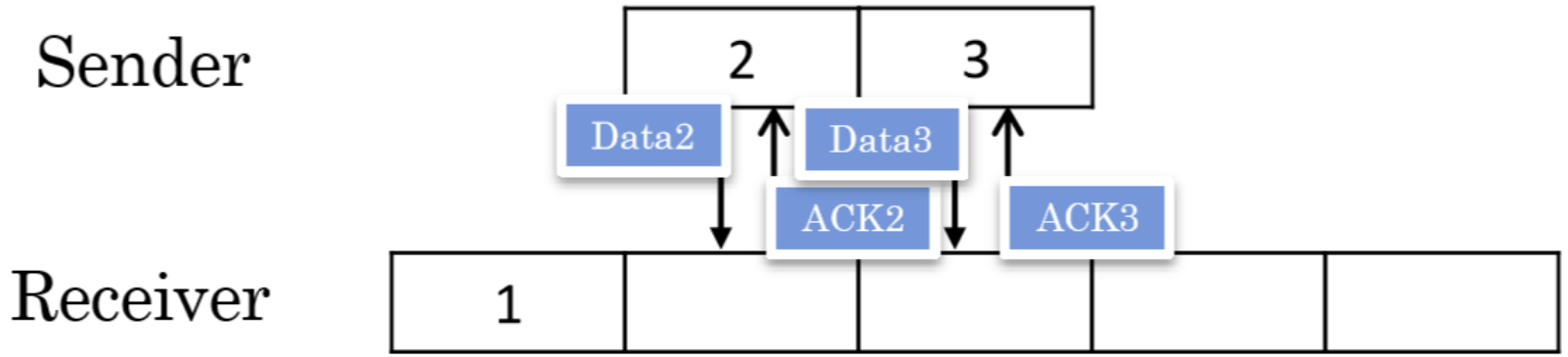
Example (*GBN + Congestion control*)

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



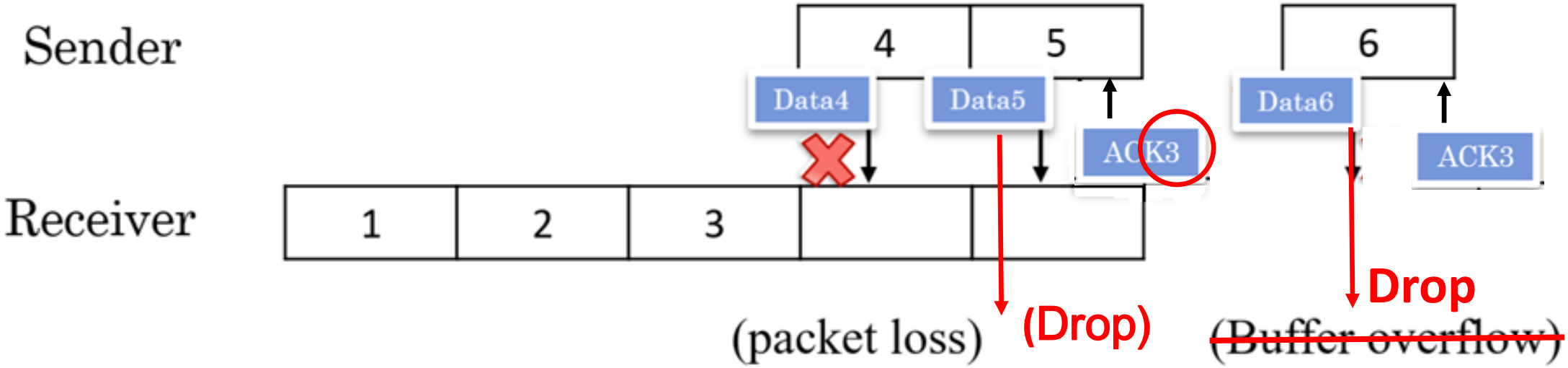
Example (*GBN + Congestion control*)

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



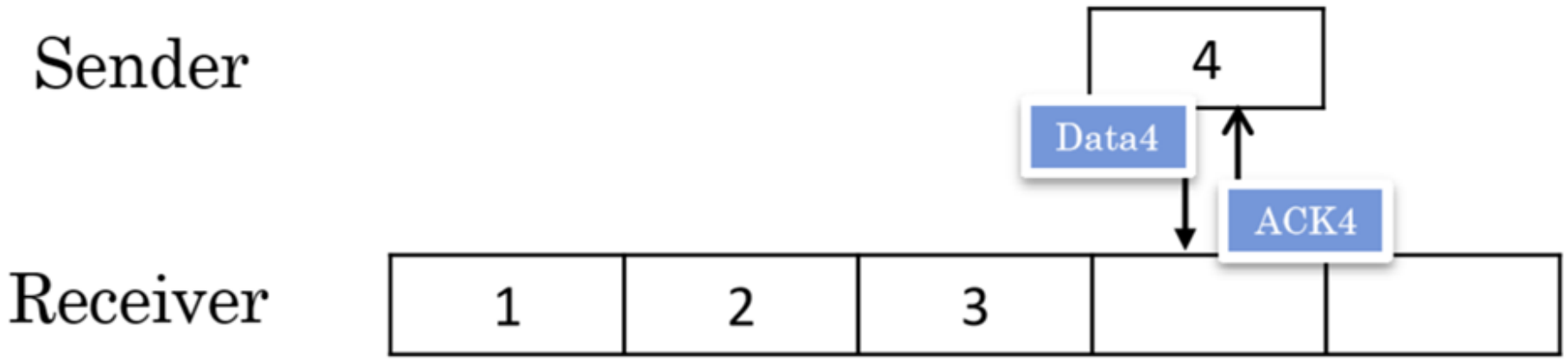
Example (*GBN + Congestion control*)

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



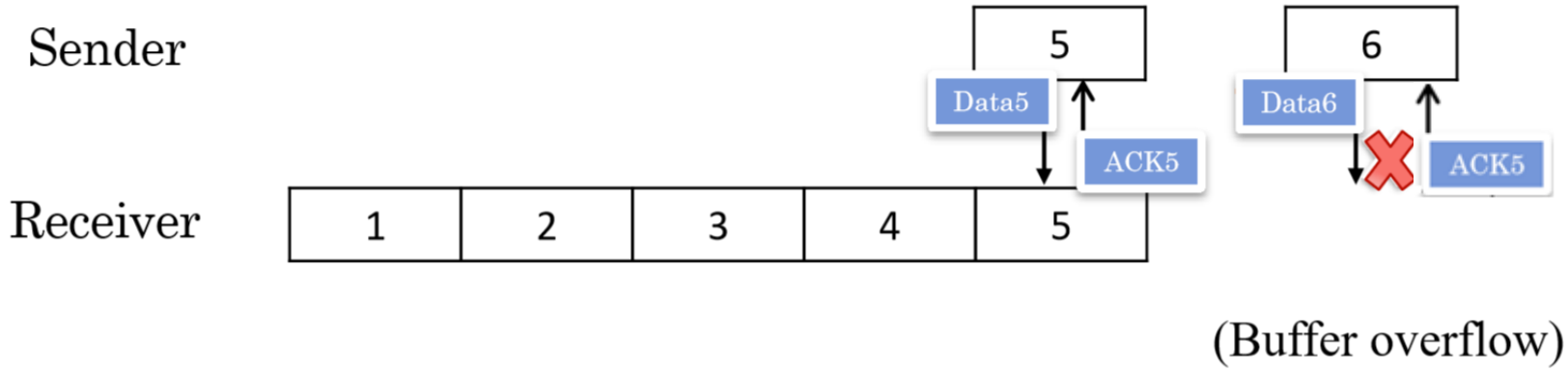
Example (*GBN + Congestion control*)

- Sender sends Data 4
- Congestion window = 1, Threshold = 1
- Receiver sends ACK 4



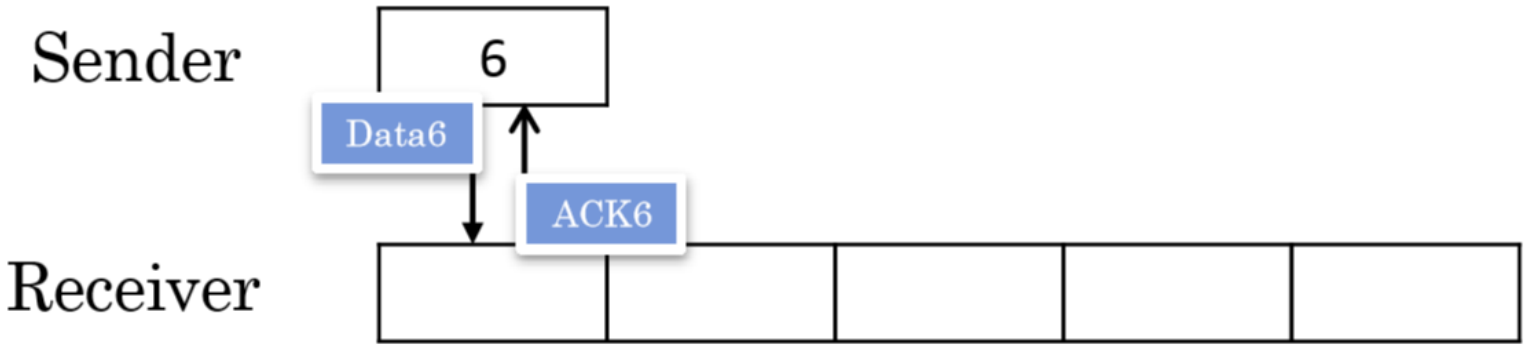
Example (*GBN + Congestion control*)

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



Example (*GBN + Congestion control*)

- Sender sends Data 6
- Congestion Window =1; Threshold =1
- Receiver sends ACK 6
- And so on…

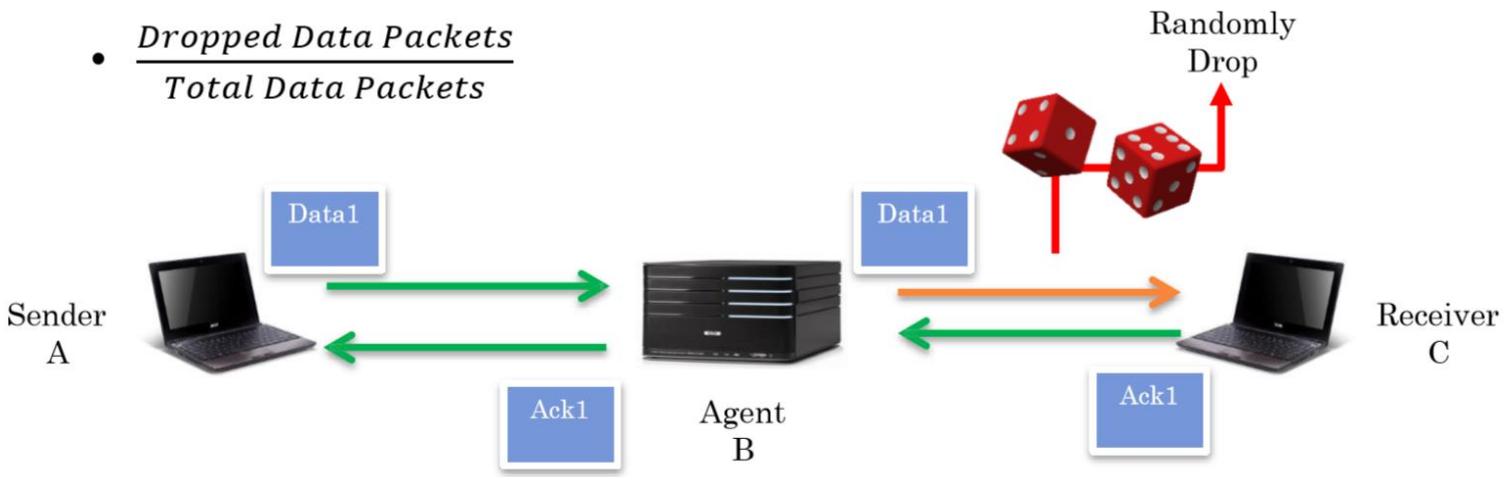


Requirement

Agent

- Forward data and ACK packets
- Randomly drop data packet [***DO NOT DROP ACK PACKETS***]
- Compute loss rate

- $\frac{Dropped\ Data\ Packets}{Total\ Data\ Packets}$



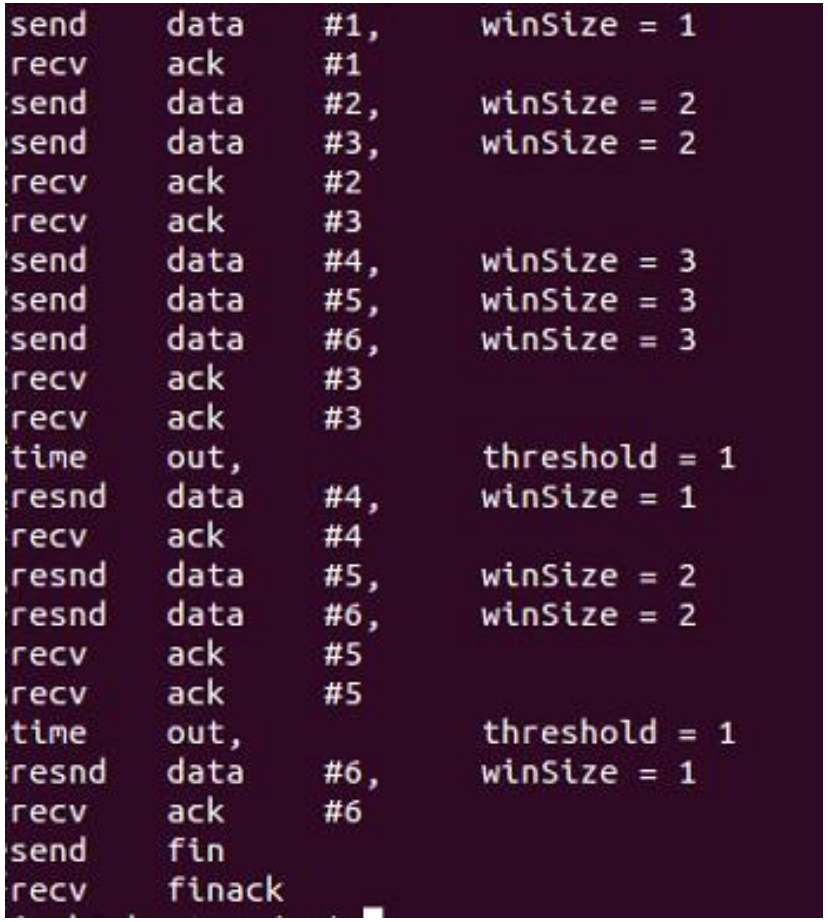
Requirement

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

Requirement

Sender



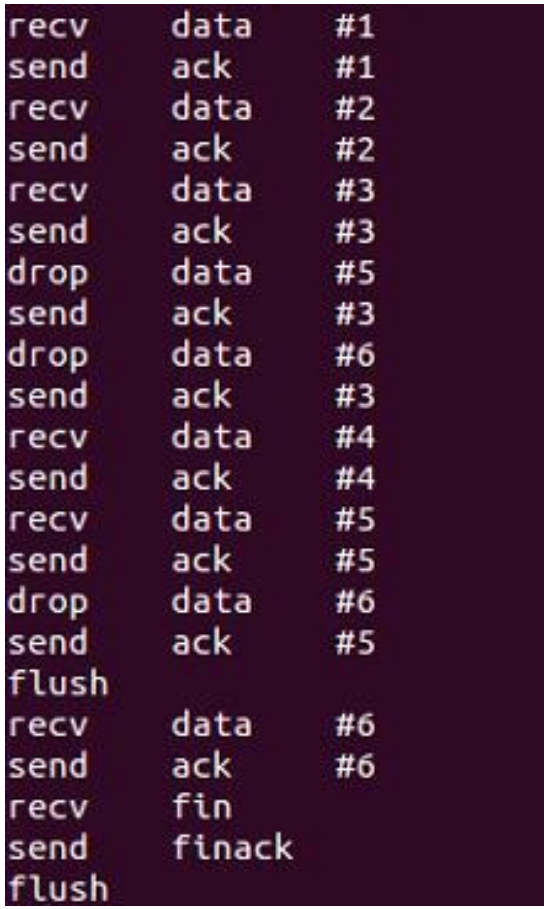
Requirement

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
```

Requirement

Receiver



Requirement

Settings

- Sender
  - Arguments: IP, Port, path of source file,… etc.
  - Default threshold:***16***
  - Input file may include media file or text file, etc.(e.g. ***./sender text.txt***)
- Receiver
  - Arguments: IP, port ,path of destination file, … etc.
  - Default buffer size: ***32***
  - Output file name: ***result.??*** (Filename Extension is the same as the input file)
- Agent
  - Arguments: IP, port, loss rate, … etc.

Requirement

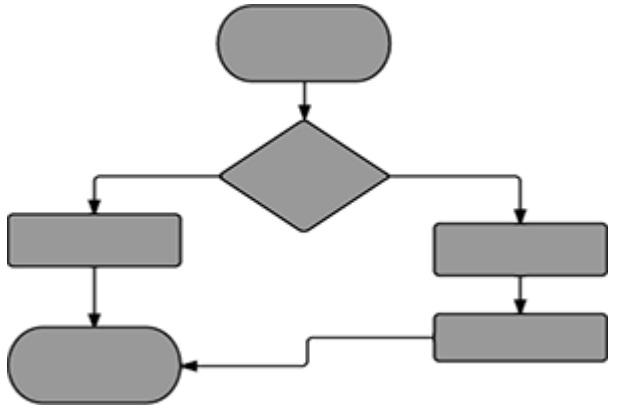
Settings

- File Size
  - More than 0.5 MB (500 KB)
- Data packet size (payload)
  - 1 KB
- Time out
  - Less than or equal to 1 sec ($\leq 1\ sec$)

Requirement

Document

- Format
  - A4, at most 2 pages
  - Digital ***PDF file only***, “HW2-Report.pdf”
- Content
  - How to execute your program
  - Explain your program structure(including ***3 flow charts*** for sender, agent, and receiver)
  - Difficulties and Solutions



Grading and Submission

Grading and Submission

Grading (100%)

Basic requirement (10%)

- Socket programming with UDP
- Language: C/C++/Python
- Without crash

Reliable transmission (20%) (page 7)

Congestion control (25%) (page 8)

Buffer handling (15%) (page 9)

Agent (10%) (page 26)

Message format (5%) (page 27)

Document (5%) (page 33)

Demo (10%) (page36)

Grading and Submission

Demo (10%)

- Please fill the demo form (will be announced on course website)
- Come to demo on time
- Discount for those are not on time
- You will get ***ZERO*** for this homework if you ***don't*** demo.

Grading and Submission

Submission Deadline

- 2017/12/15 (Fri.) 23:59 (UTC+8)
- Late submission: 20% off per day
- ***NOT*** accept after 23:59, 12/17, 2017

Naming

- [Student ID].zip Ex: r069xxxxx.zip
- Email subject: [CN2017] Homework2_studentID
- Email: ntu.cnta@gmail.com