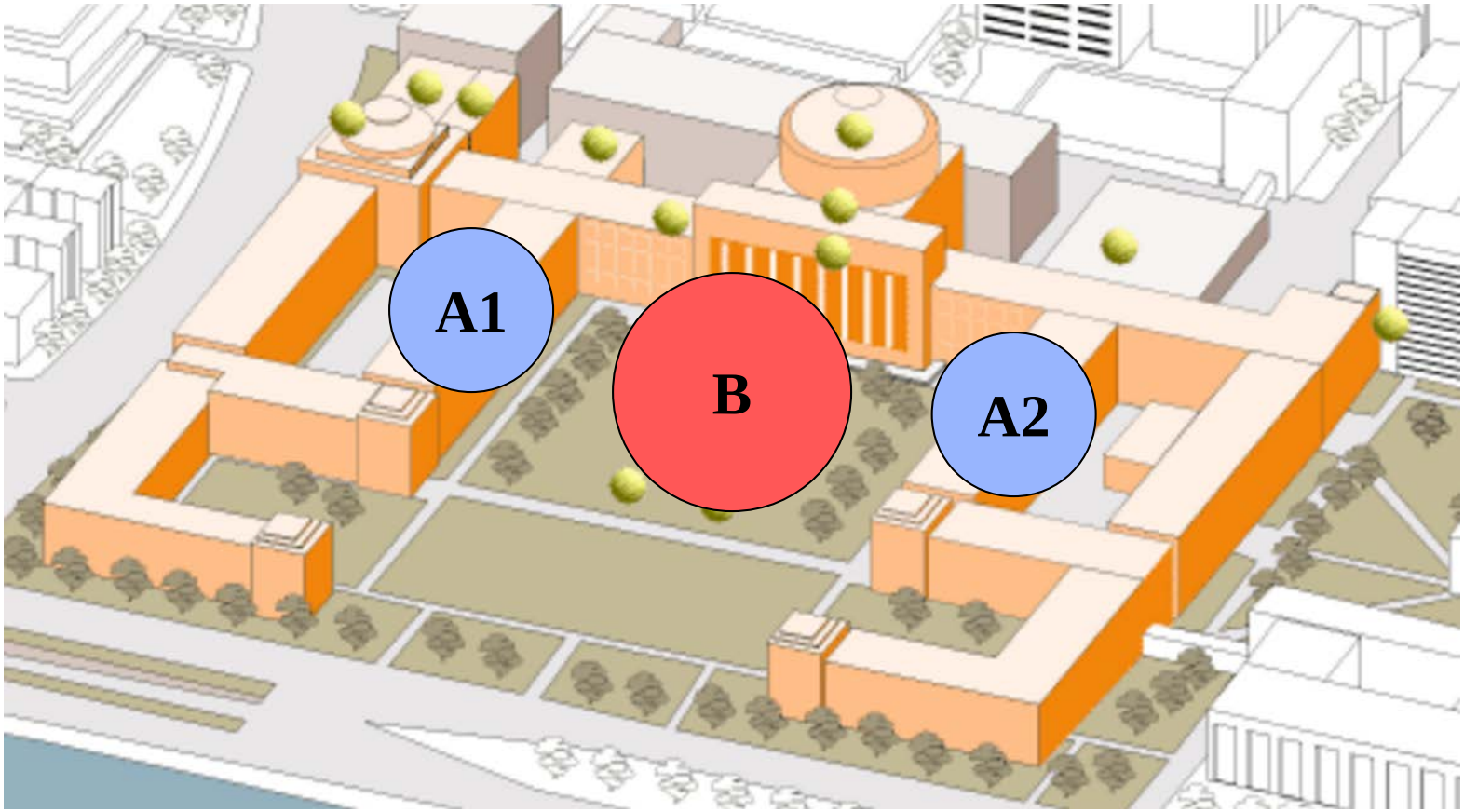
16.36: Communication Systems and Networks

Lecture 19 - The Data Link Layer: Automatic Repeat Request Protocols

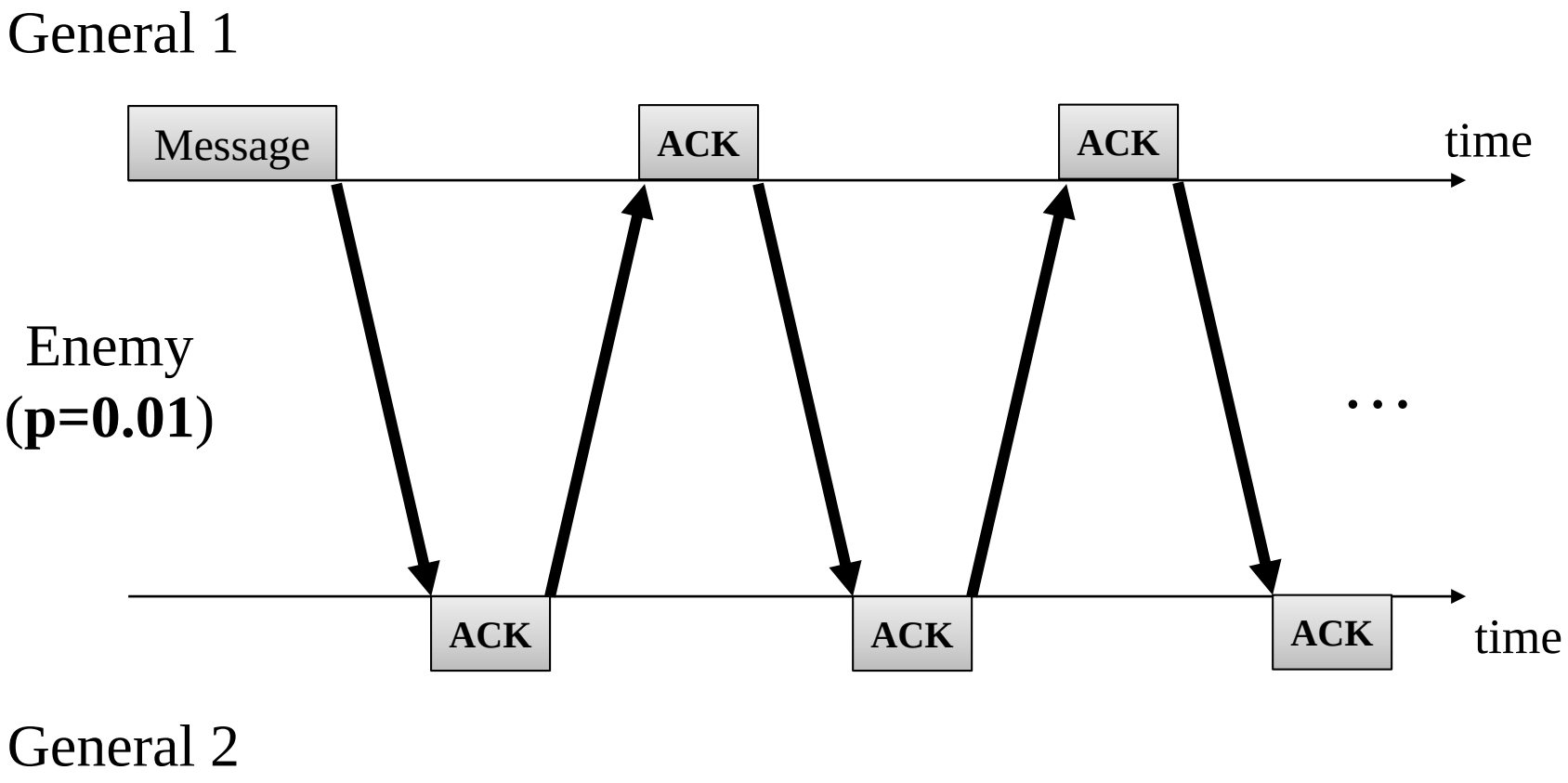
Igor Kadota and Eytan Modiano
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology

Cambridge, May 2, 2019

Two Generals' Problem

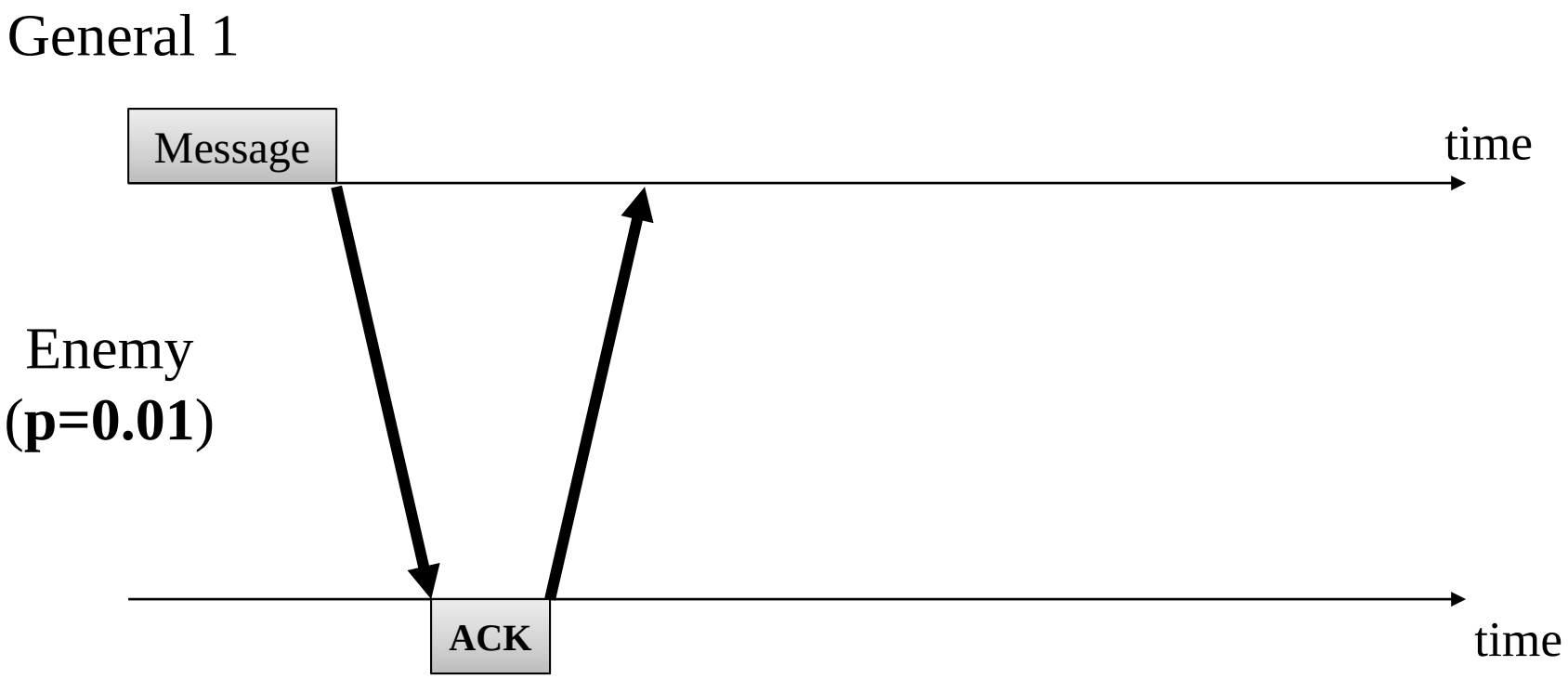


Two Generals' Problem



Infinite handshakes are needed for both Generals to be sure that a consensus was reached.

One General Problem



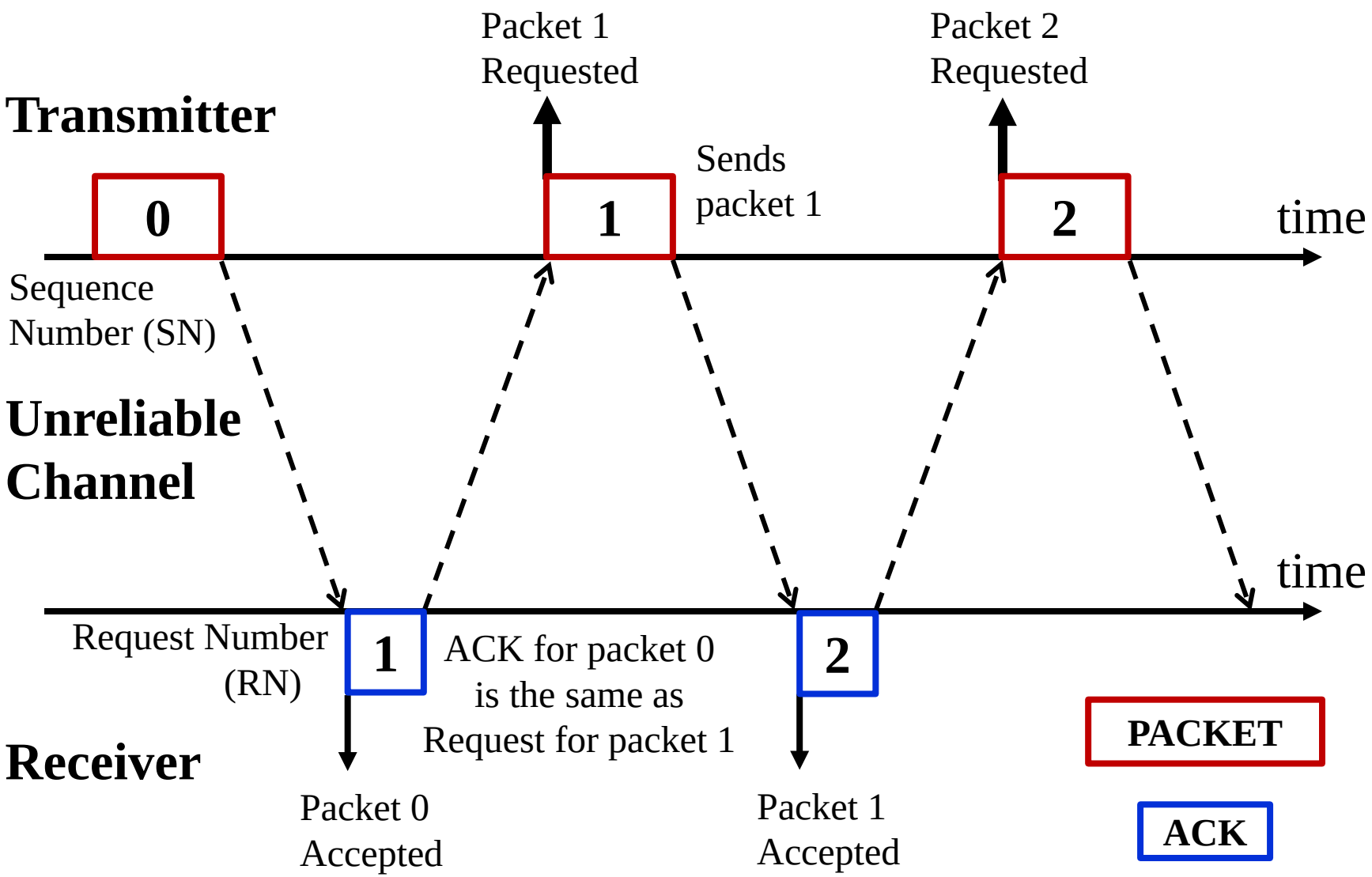
Only 2 handshakes are needed for General 1
to be sure about the reception of the message.

Automatic Repeat ReQuest (ARQ)

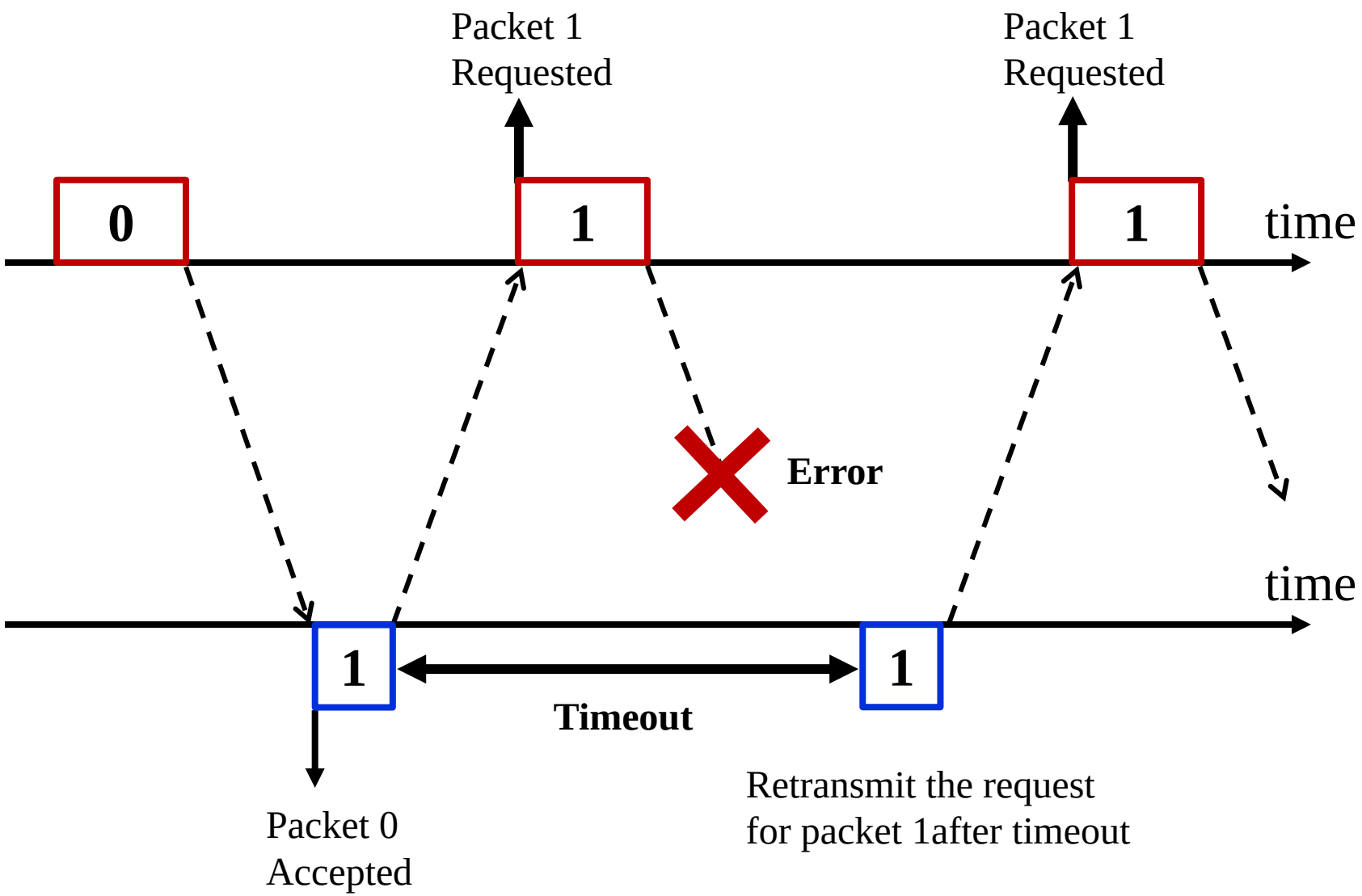
- Transmission with ACK
- Error detection (CRC)
- Error recovery with ARQ:
  - request retransmission of packets received with error
- **Three common schemes**
  - Stop & Wait
  - Go Back N
  - Selective Repeat Protocol (SRP)

ARQ Protocols: Stop & Wait

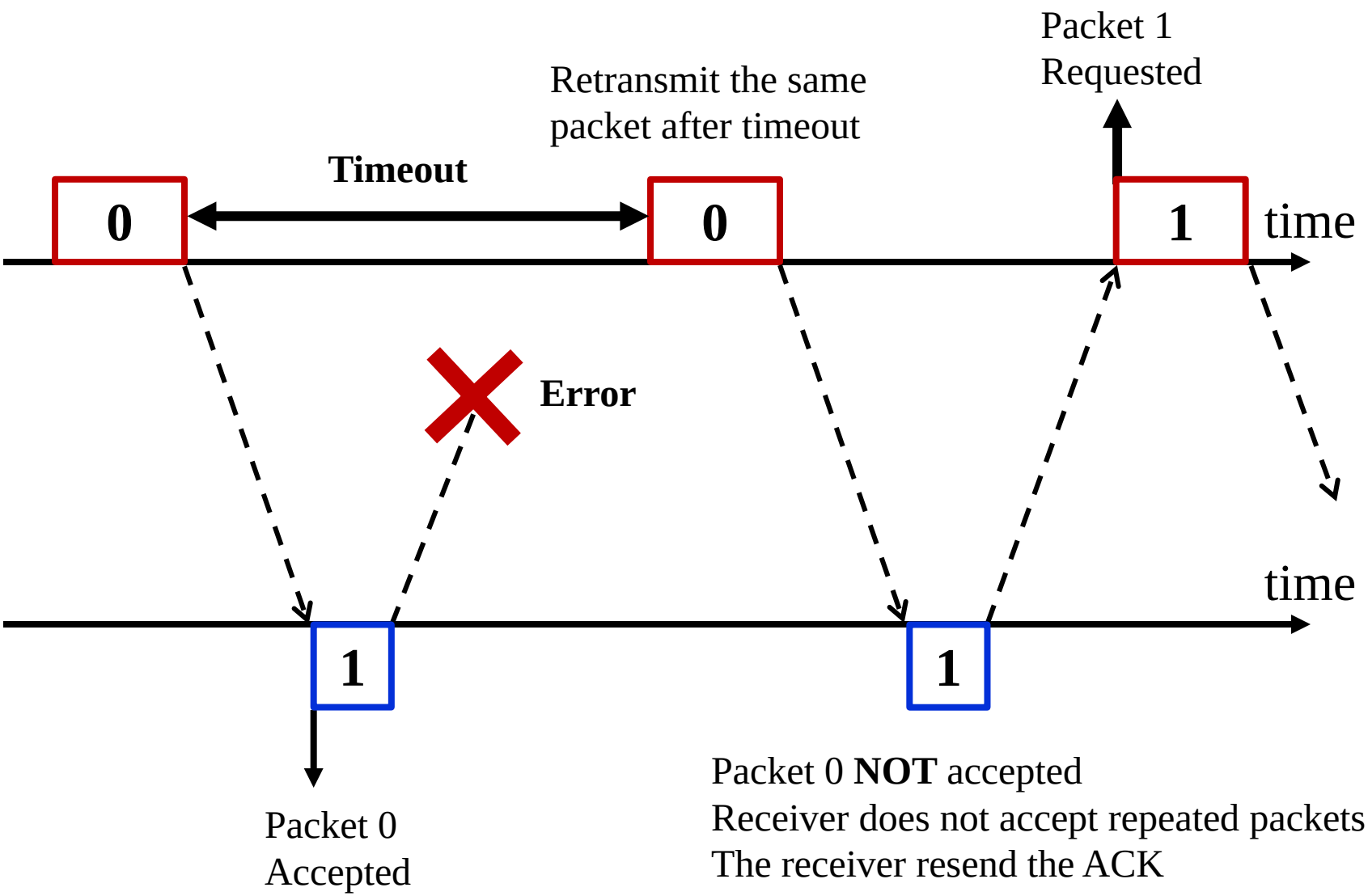
Stop and Wait Protocol



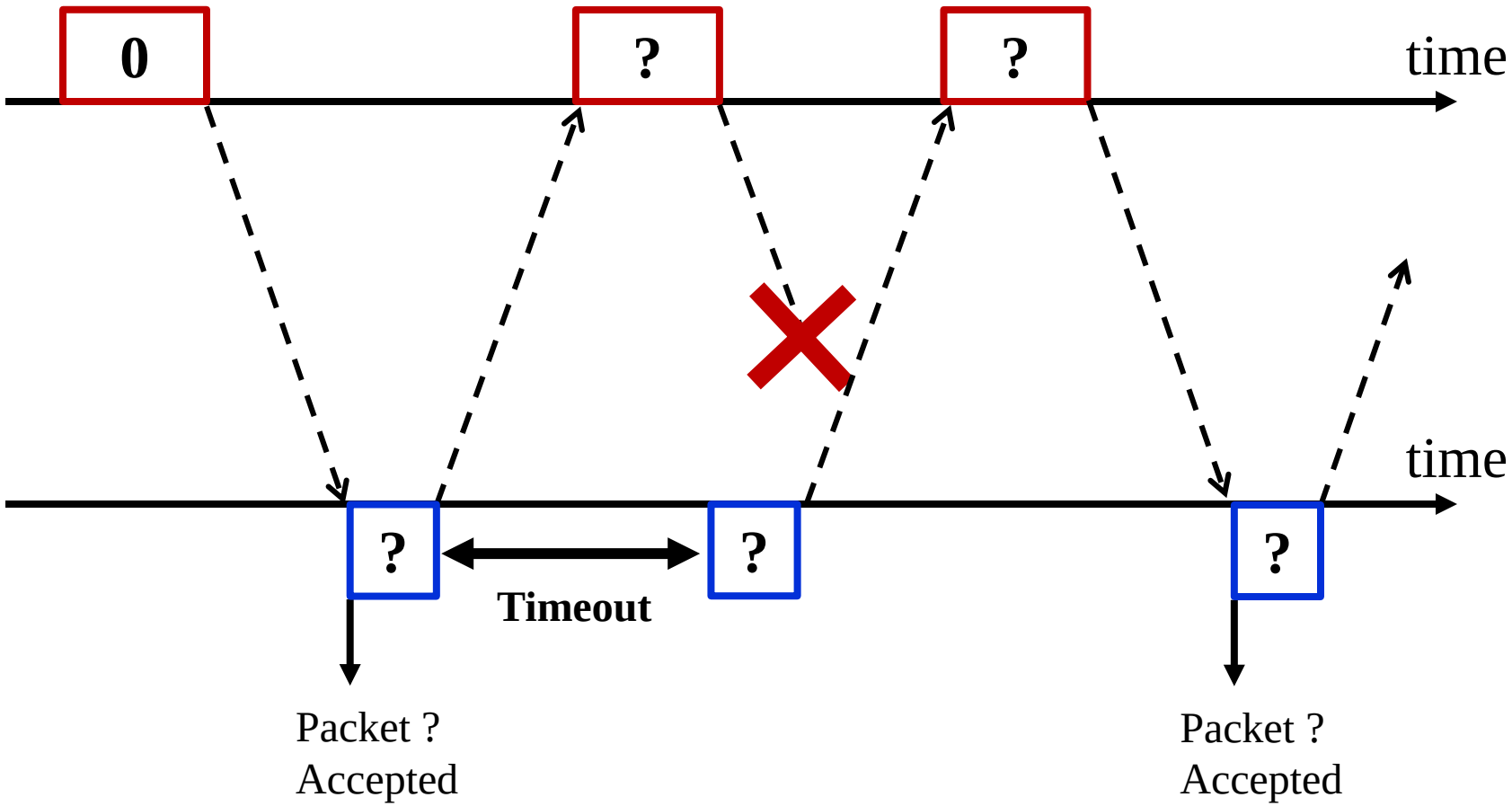
Stop and Wait Protocol



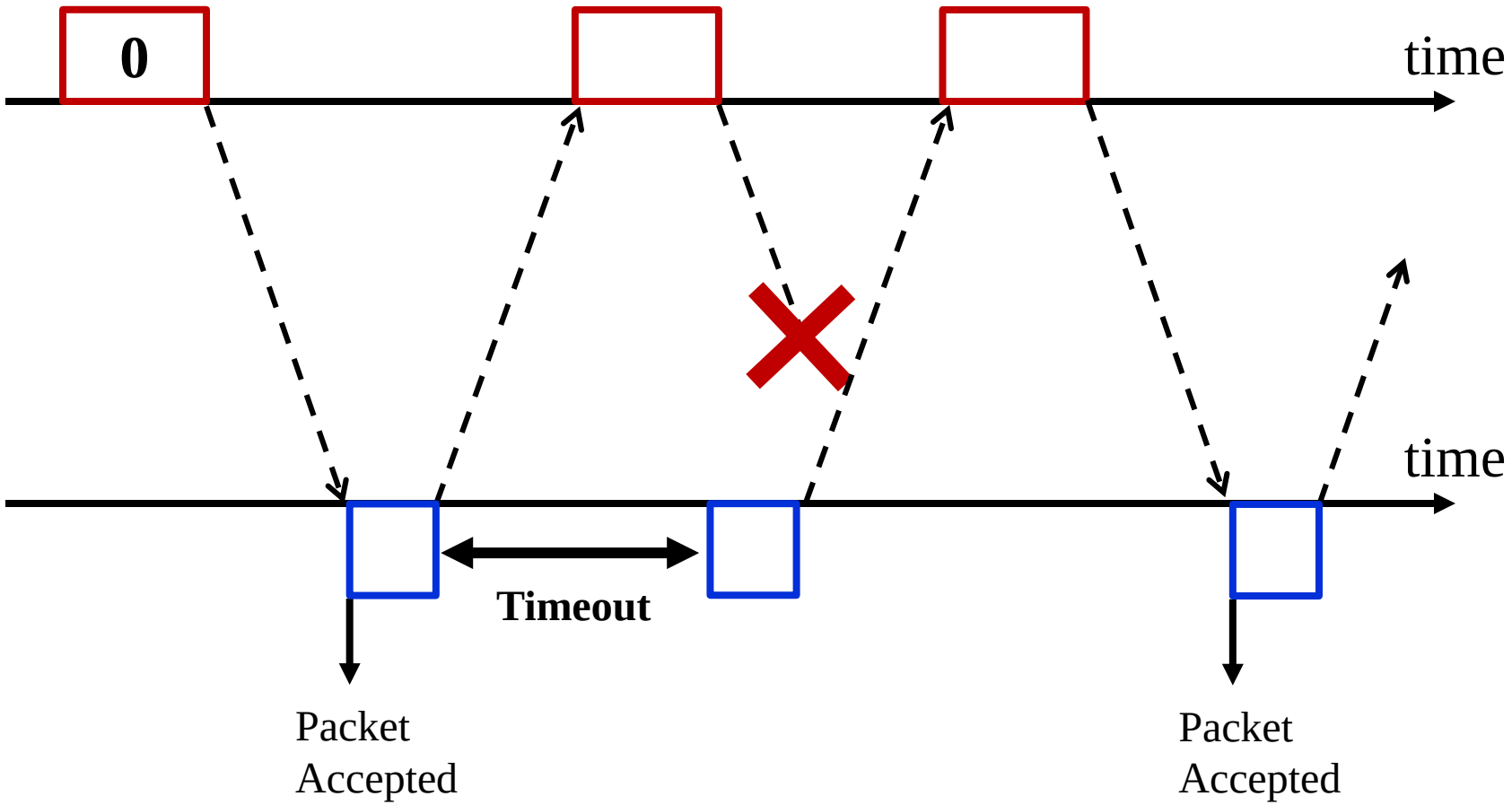
Stop and Wait Protocol



Stop and Wait Protocol: Exercise



Stop and Wait Protocol: Exercise



Stop and Wait Protocol - Algorithm

Algorithm at **sender**:
(with initial condition SN=0)

1) Accept packet from higher layer when available;
Then, assign number SN to the packet

2) Transmit packet SN

3) Wait for an error free ACK from B

i. if received and it contains RN>SN in the request # field, set SN to RN and go to 1

ii. if not received within given timeout (TO), go to 2

Stop and Wait Protocol - Algorithm

Algorithm at **receiver**:
(with initial condition RN=0)

1) Whenever an error-free packet is received from A with a SN equal to RN, release received packet to higher layer and increment RN.

2) At arbitrary times, but within bounded delay after receiving any error_free frame from A, transmit a frame to A containing RN in the request # field.

Stop and Wait Protocol

- Packet numbering:
  - Sender: Sequence Numbers (SN)
  - Receiver: Request Numbers (RN)

    RN = j is the same as an ACK for packet j-1

- Sender transmits **one packet at a time** and waits for the respective ACK
  - Idle while waiting for ACK
  - Retransmits the packet after a timeout (TO)

- Receiver only ACKnowledges the packet it requested
  - Retransmits the last ACK after a timeout (TO)

- **Does this mechanism work?**

Correctness of Stop and Wait

- Informal proof that Stop and Wait is correct.

- **Correct**: a never-ending stream of packets can be delivered to B, **in order and without repetitions or deletions.**

- Proof is break into two parts: **safety** and **liveness**.
  - Safe: algorithm never produces an incorrect result
  - Live: it can continue forever (no deadlock)

- Assumptions:
  - Errors are detected by the CRC;
  - There is a probability $q > 0$ that each frame is received with no errors;
  - Link is initially empty;
  - Transmitter has SN=0 and receiver is awaiting the packet with SN=0.

Correctness of Stop and Wait

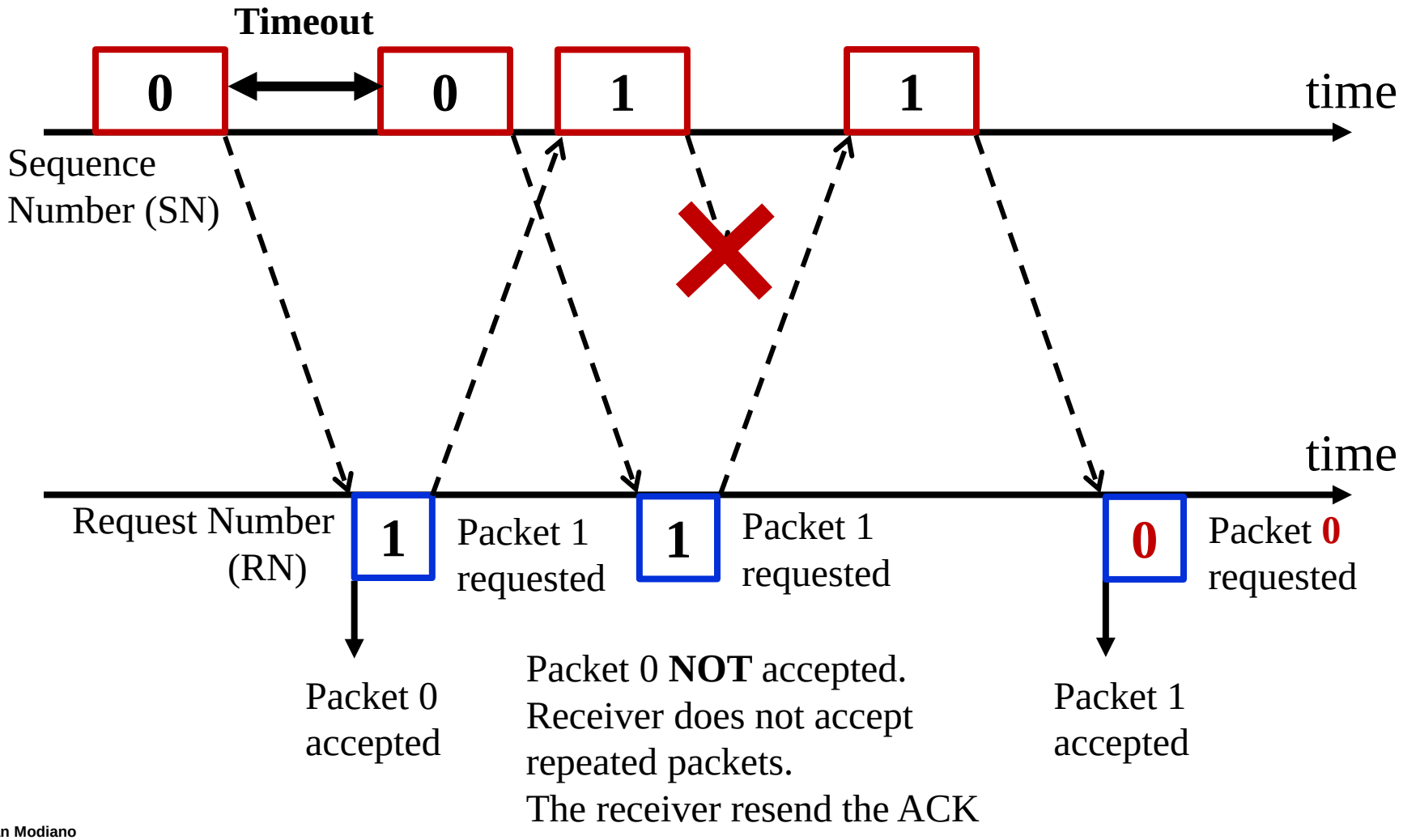
- SAFETY: show that no packet is ever released out of order or more than once
  - Start with packet ‘0’ and receiver does not increment its RN until ‘0’ is received; upon which it can only accept packet ‘1’, etc.

- LIVENESS: show that every packet is eventually released
  - The sender keeps sending a packet until it gets an ACK, so eventually every packet is correctly received

Note on packet numbering: packets are numbered modulo 2 (0 or 1)

- At any time, the received packet can only be RN or RN-1.
- Mod 2 numbering can be used to distinguish between these packets.

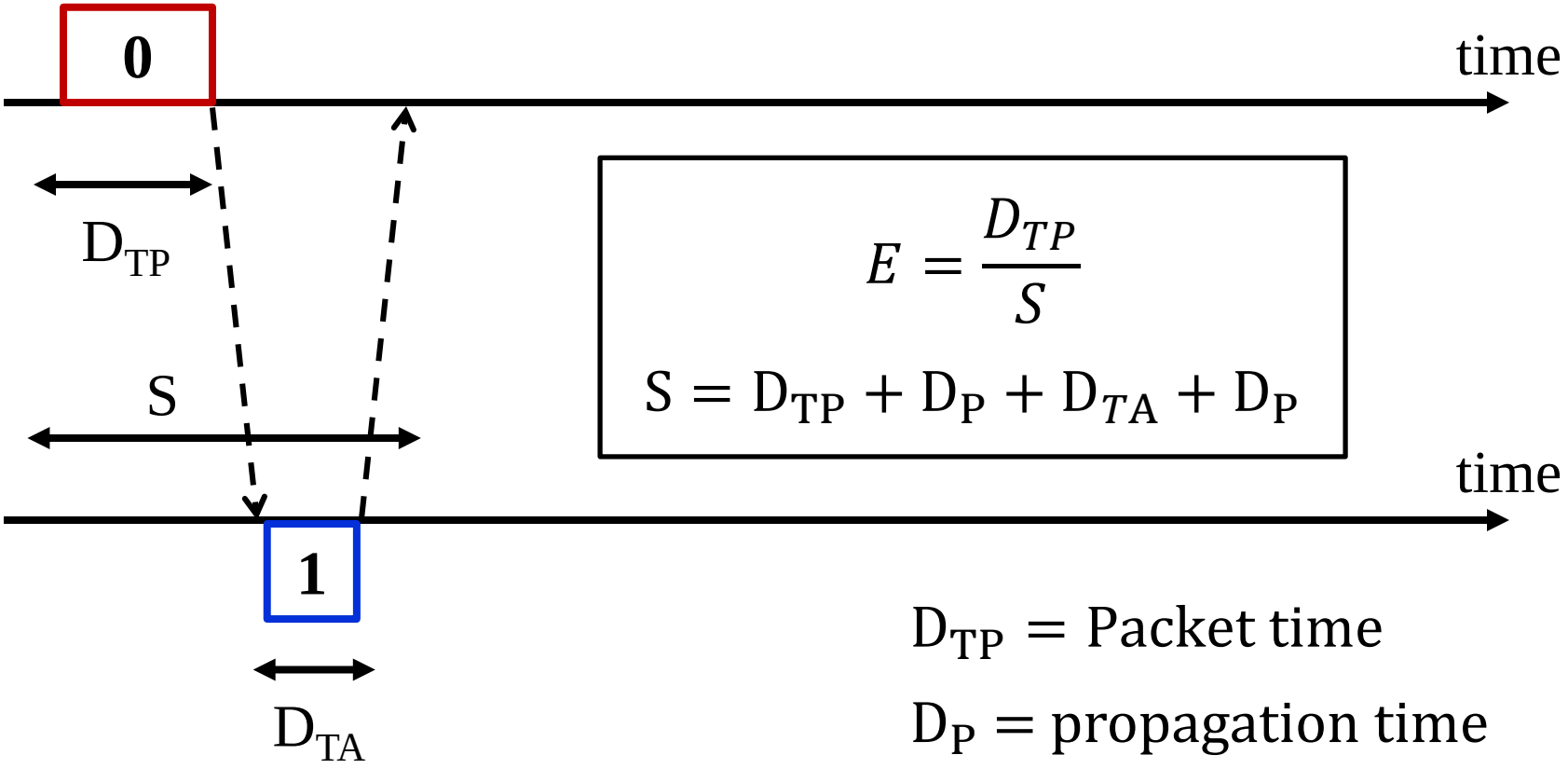
Numbering – Mod 2



Efficiency of Stop and Wait

High efficiency

When? Example?



$$E = \frac{D_{TP}}{S}$$

$$S = D_{TP} + D_P + D_{TA} + D_P$$

D_{TP} = Packet time

D_P = propagation time

D_{TA} = ACK time

Efficiency of Stop and Wait – Examples

- **56kbps modem**
  - Low propagation delay environment, distance = 5 miles ~ 8km
    $D_p = 8km / 3x10^8 = 27\mu s$
  - R = 56,000bps; Packet length = 1000 bits; ACK= 50 bits
    $\Rightarrow D_{TP} = 17857\ \mu s$; $D_{TA} = 893\ \mu s$

  S = 18804 μs ⇒ Efficiency = 17857/18804 ~ 95%

- **High Speed Metropolitan Area Network**
  - R = 100Mbps, distance = 100 miles ~ 160km

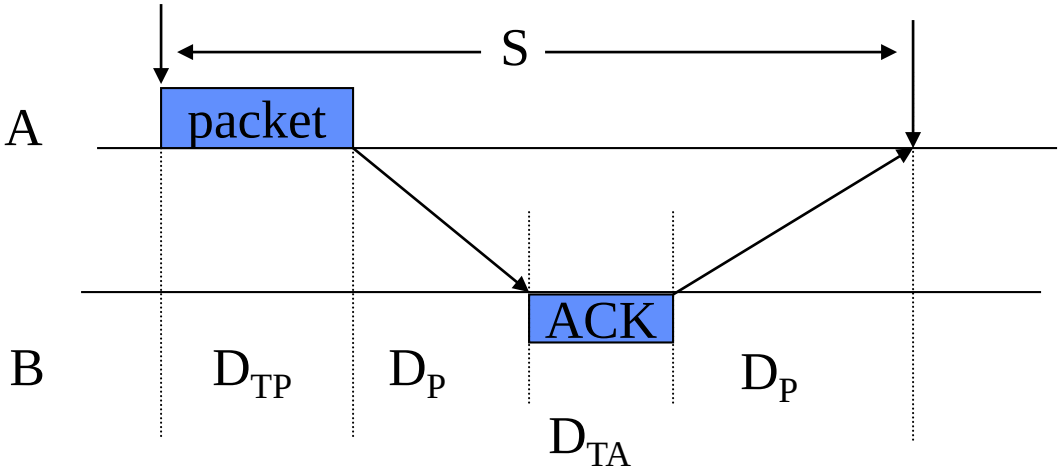
  ⇒ $D_{TP} = 10^{-5}$ seconds, $D_{TA} = 5x10^{-7}$ seconds, $D_p = 5x10^{-4}$ seconds

  S = 10^{-3} seconds ⇒ Efficiency = $10^{-5}/10^{-3}$ = 1%

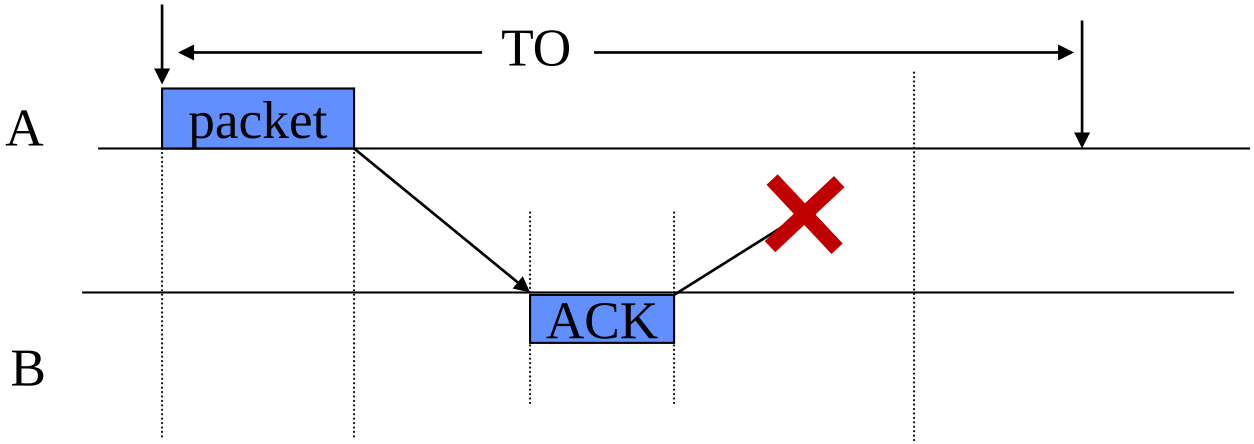
Efficiency of Stop and Wait with Errors

Let **P** = the probability of an **error** in the transmission of a packet or in its acknowledgment

No Error:



Error:



Efficiency of Stop and Wait with Errors

Let **P** = the probability of an **error** in the transmission of a packet or in its acknowledgment

$S = D_{TP} + 2D_P + D_{TA}$

TO = the timeout interval

X = the amount of time that it takes to transmit a packet and receive its ACK. This time accounts for retransmissions due to errors

$$\mathbb{E}[X] = S + TO * \mathbb{E}[\#\text{ of errors}]$$

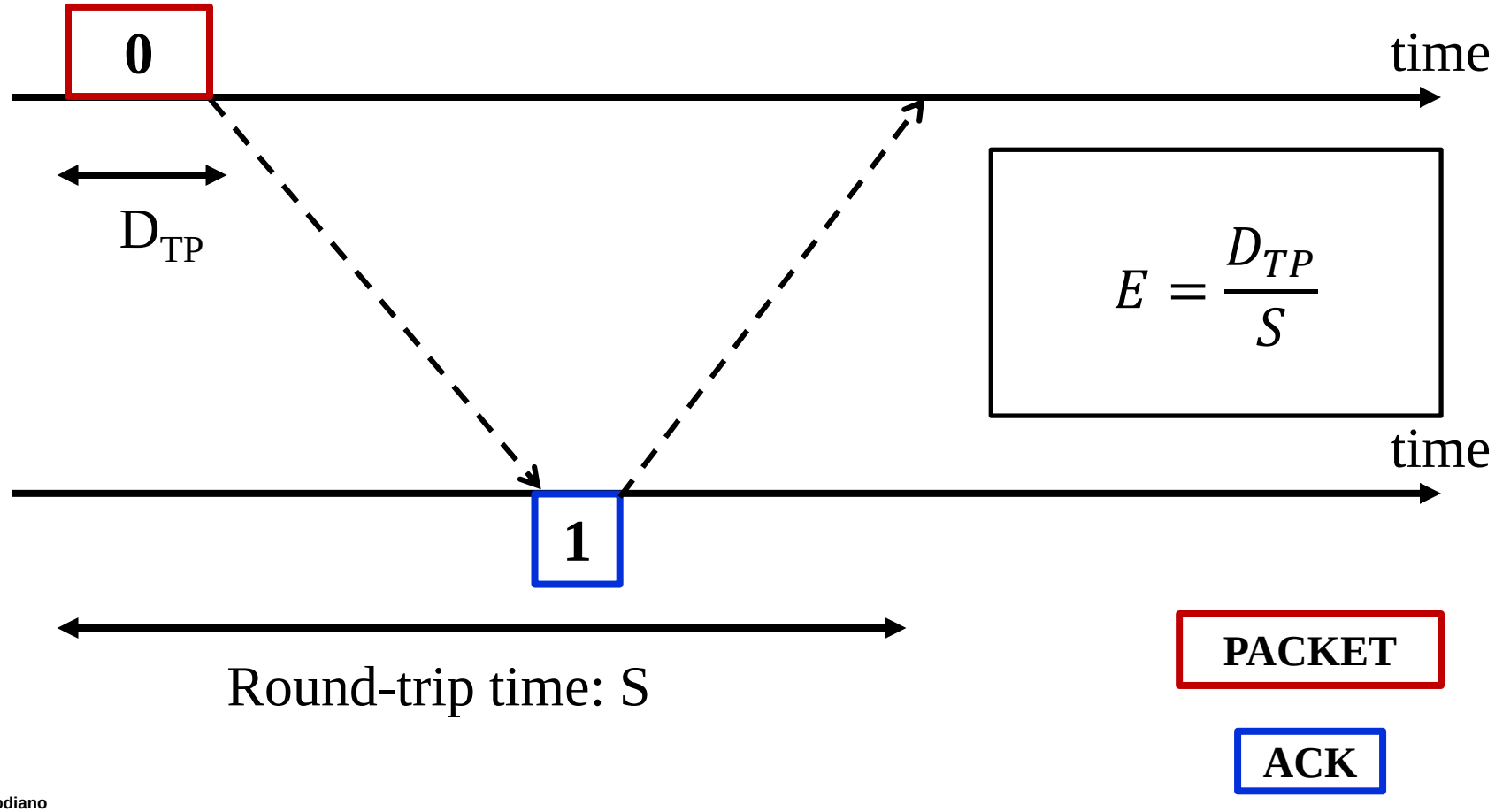
$$\mathbb{E}[X] = S + TO * P/(1-P)$$

$$\text{Efficiency} = D_{TP}/\mathbb{E}[X]$$

Efficiency of Stop and Wait

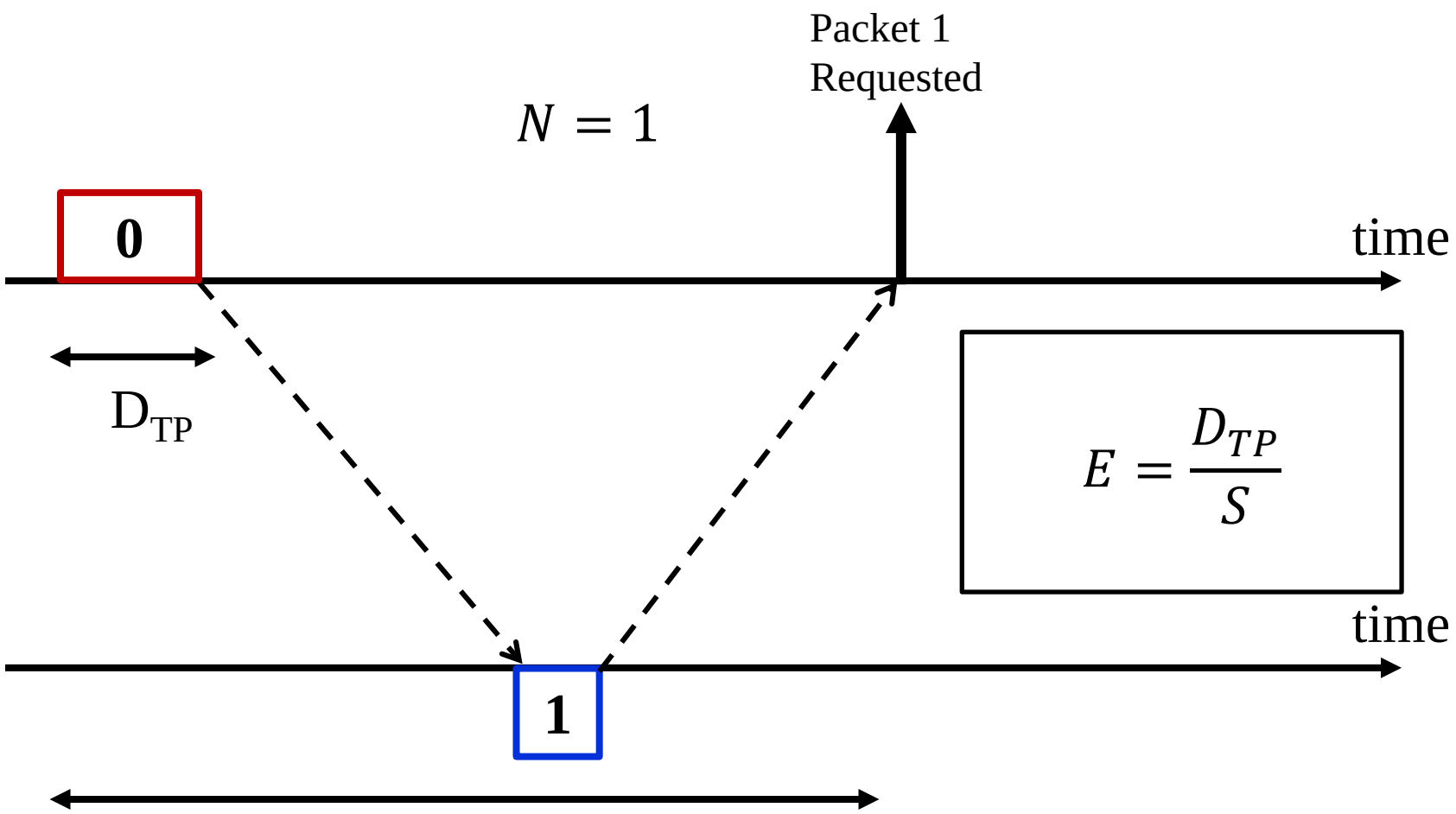
Low efficiency

How to solve?



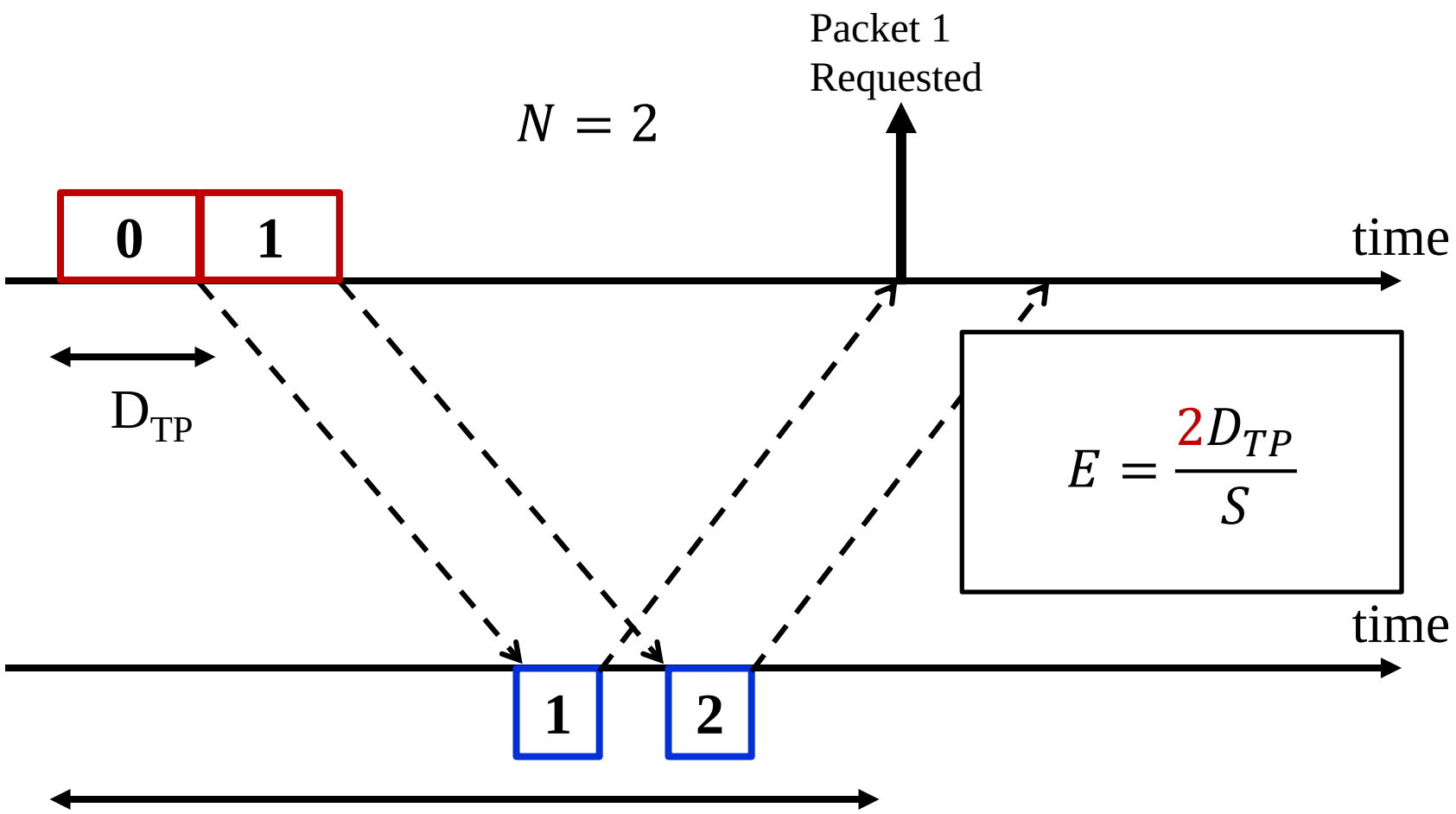
ARQ Protocols: Go Back-N

Go Back N – Efficiency



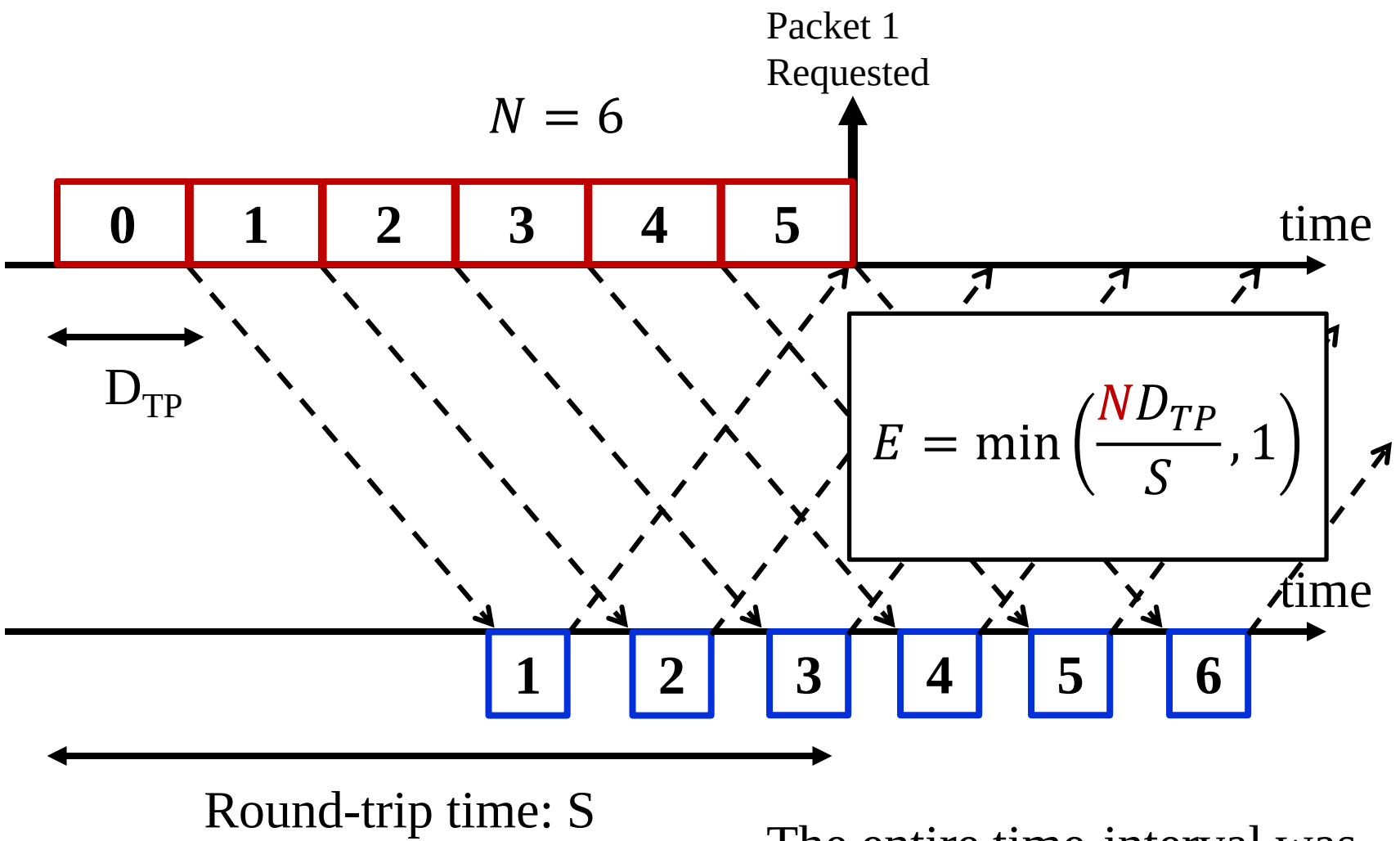
Supposing there are no transmission errors

Go Back N – Efficiency



Supposing there are no transmission errors

Go Back N – Efficiency



The entire time-interval was utilized for transmitting data

Go Back N – Sliding Window

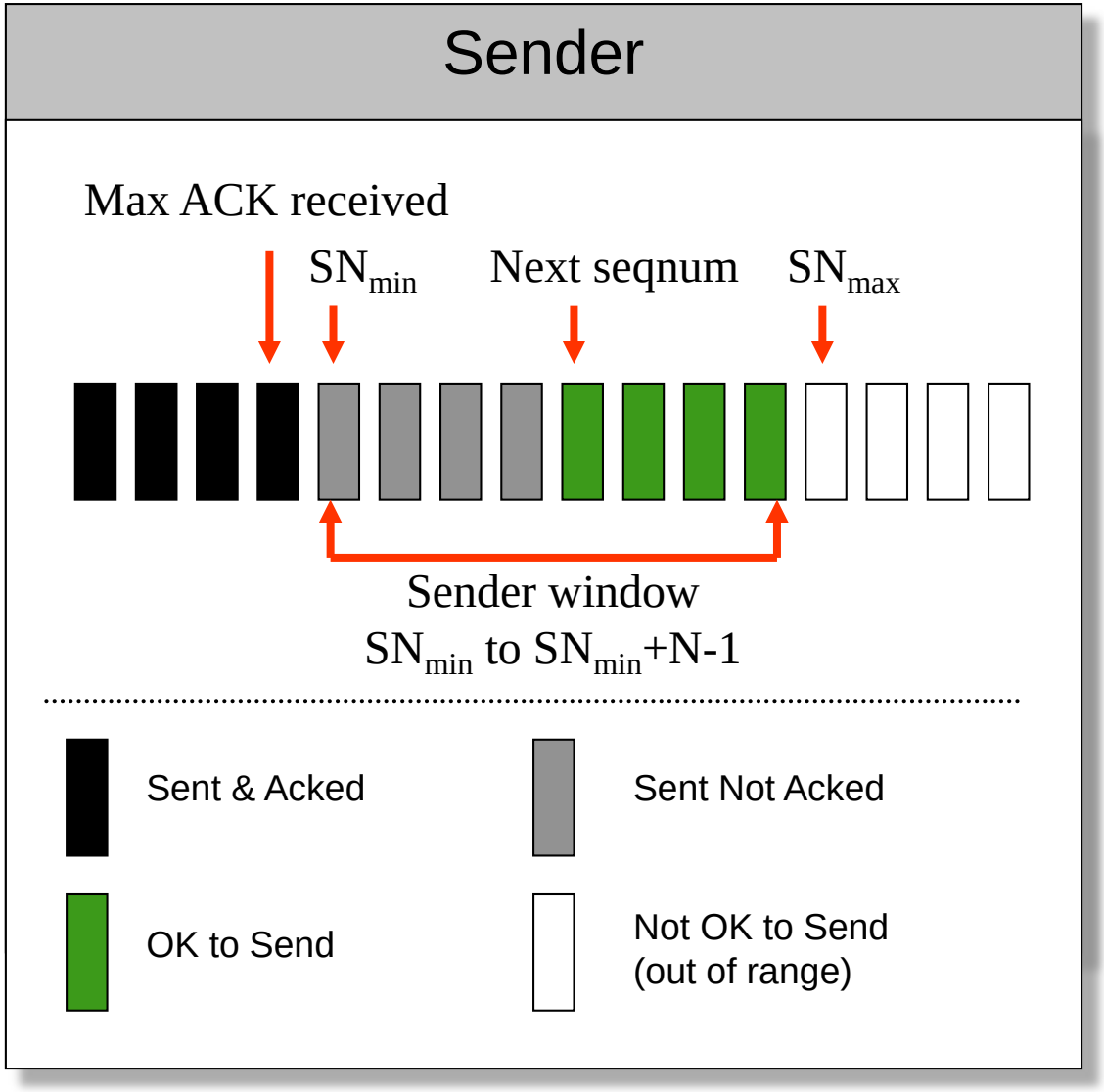
- Go Back N or Sliding Window Protocol

- Window size = N
  - Sender can transmit up to N packets without ACKnowledgement
  - Packet i+N cannot be transmitted **before receiving the ACK** with RN=i+1
  - Window ranges from the last value of RN obtained from the receiver (denoted SN_{min}) to SN_{min}+N-1

Receiver does NOT change. Receiver operates like in Stop and Wait.

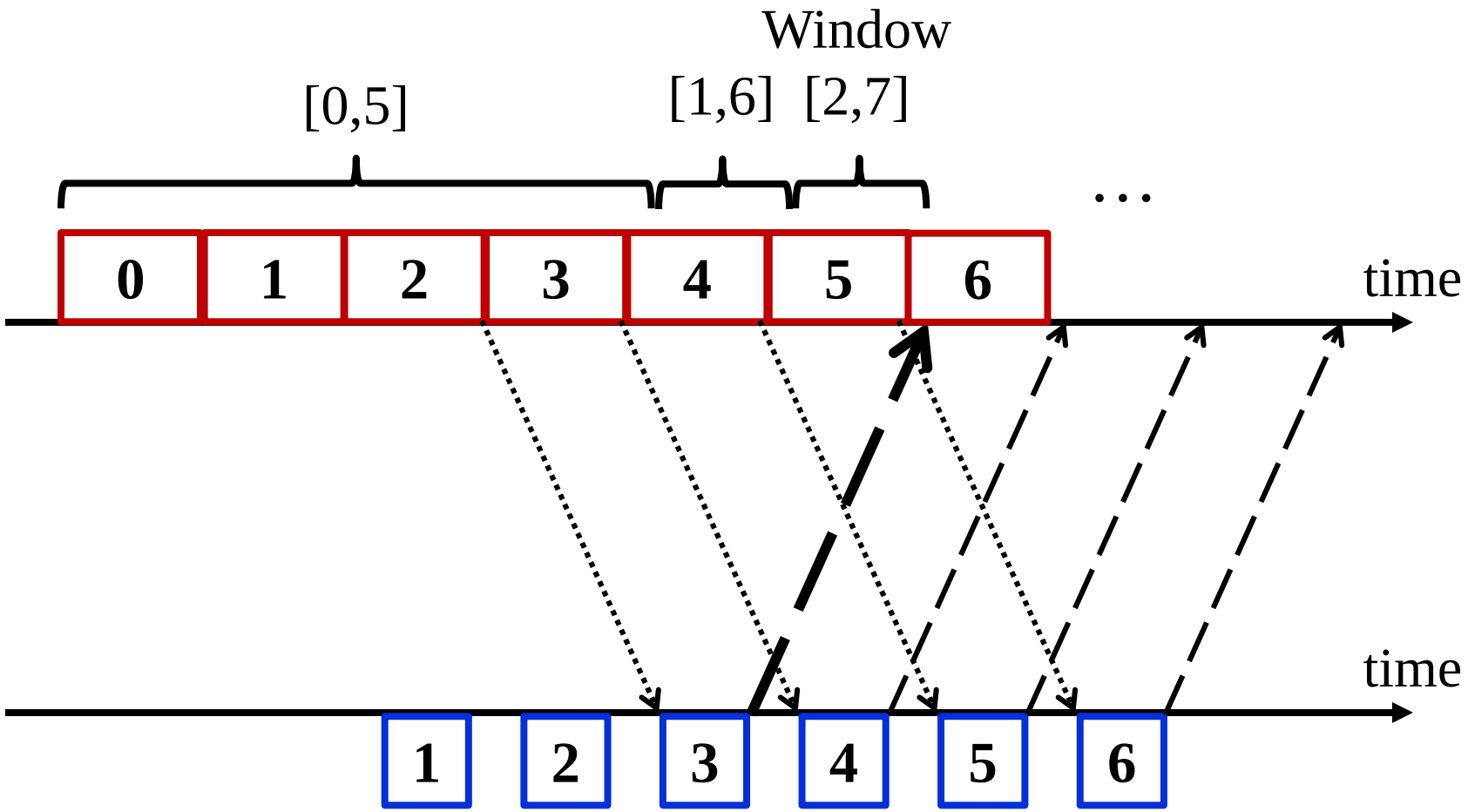
- Receive packets **in order.** Accepts ONLY the packet with SN=RN
- Send RN = RN + 1 ⇒ ACK for all packets up to and including i

Go Back N – Sliding Window



Go Back N – Sliding Window – Example

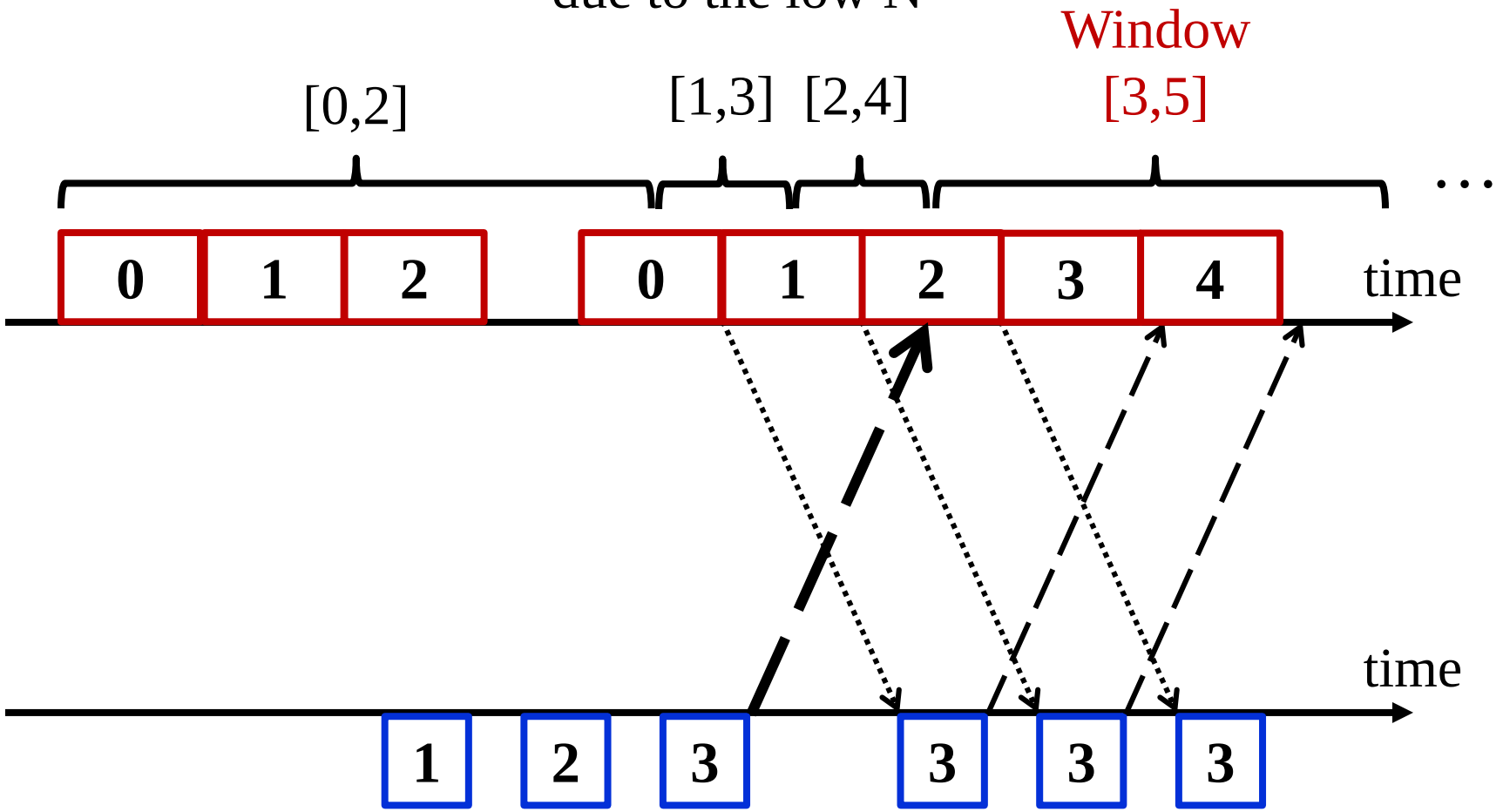
For $N = 6$ High efficiency when no error occurs



Go Back N – Sliding Window – Example

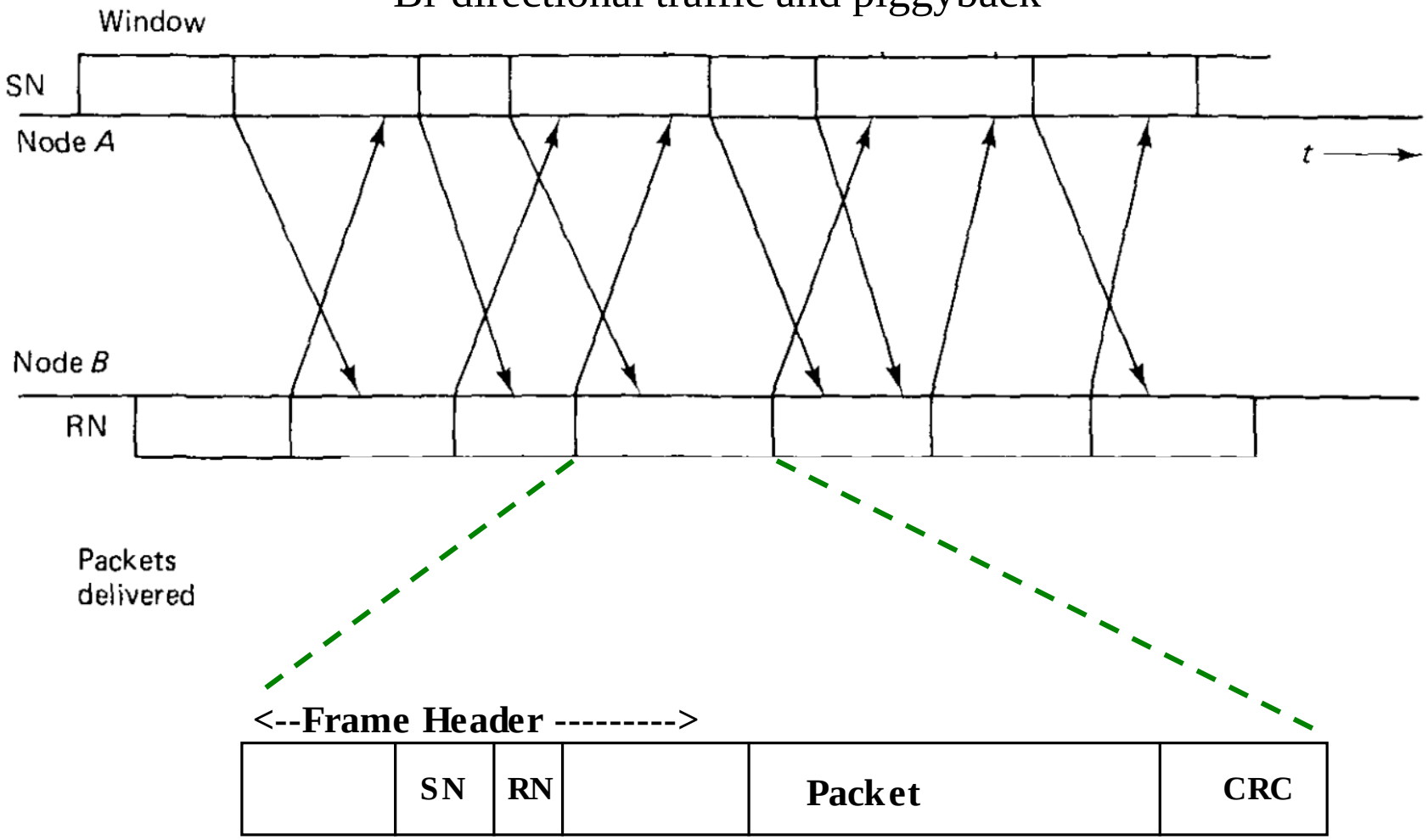
For $N = 3$

Had to retransmit the entire window due to the low N

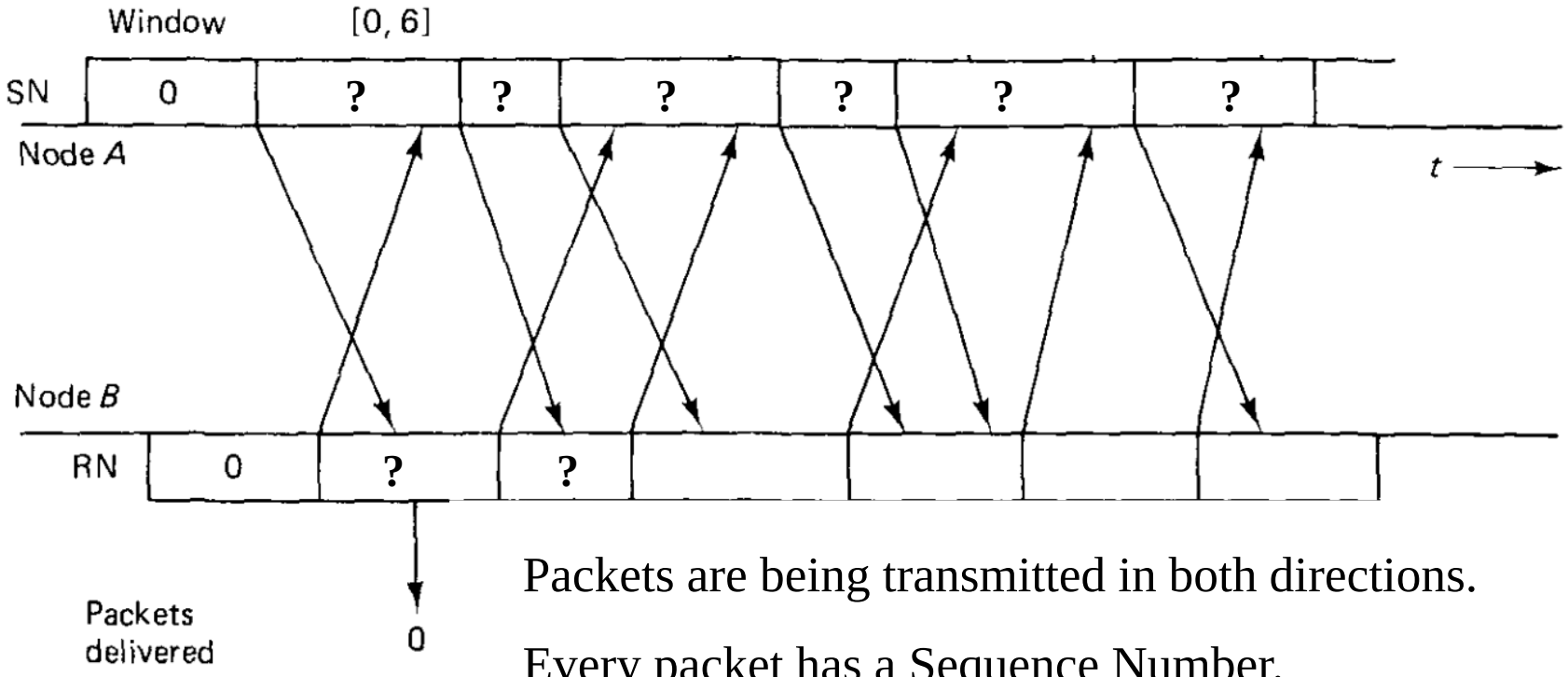


Example of Go Back 7 ARQ

Bi-directional traffic and piggyback



Example of Go Back 7 ARQ



Packets are being transmitted in both directions.

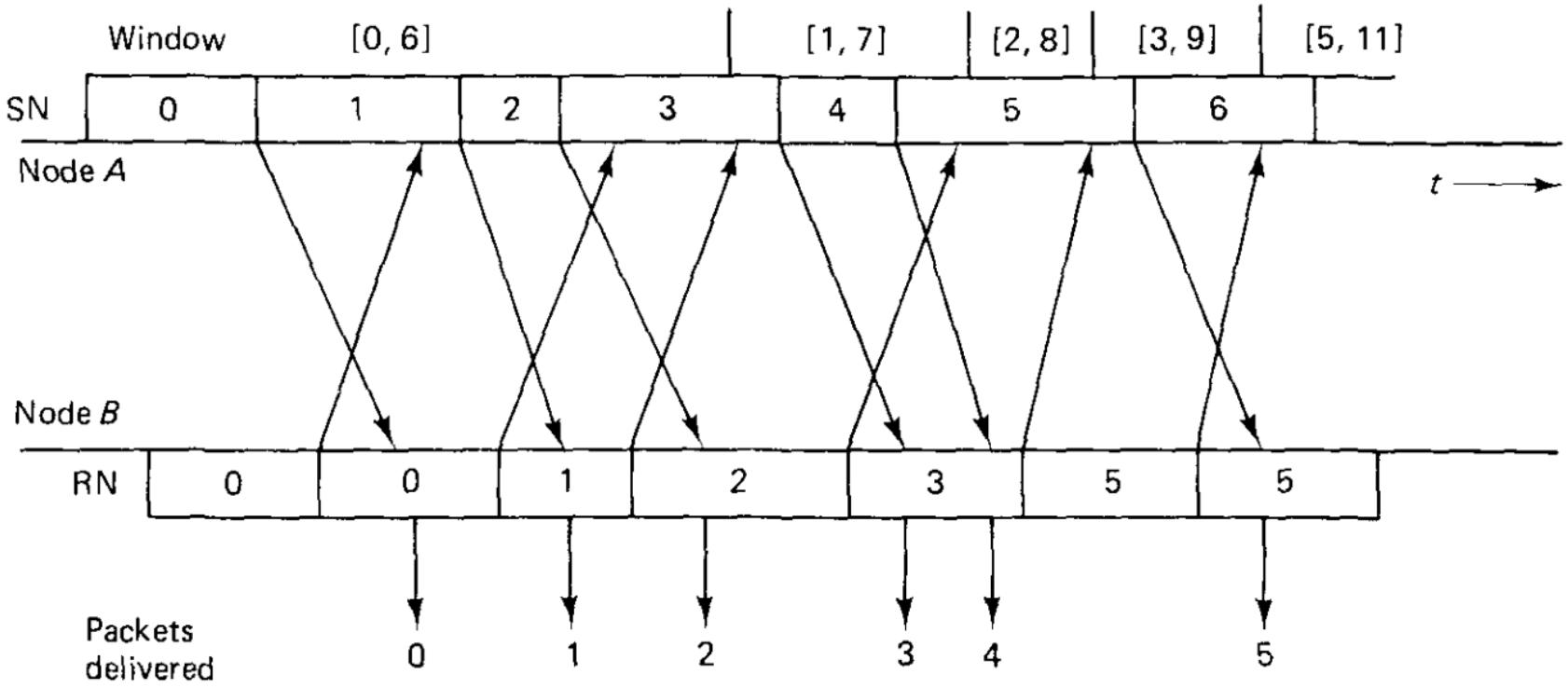
Every packet has a Sequence Number.

Request Numbers are piggybacked.

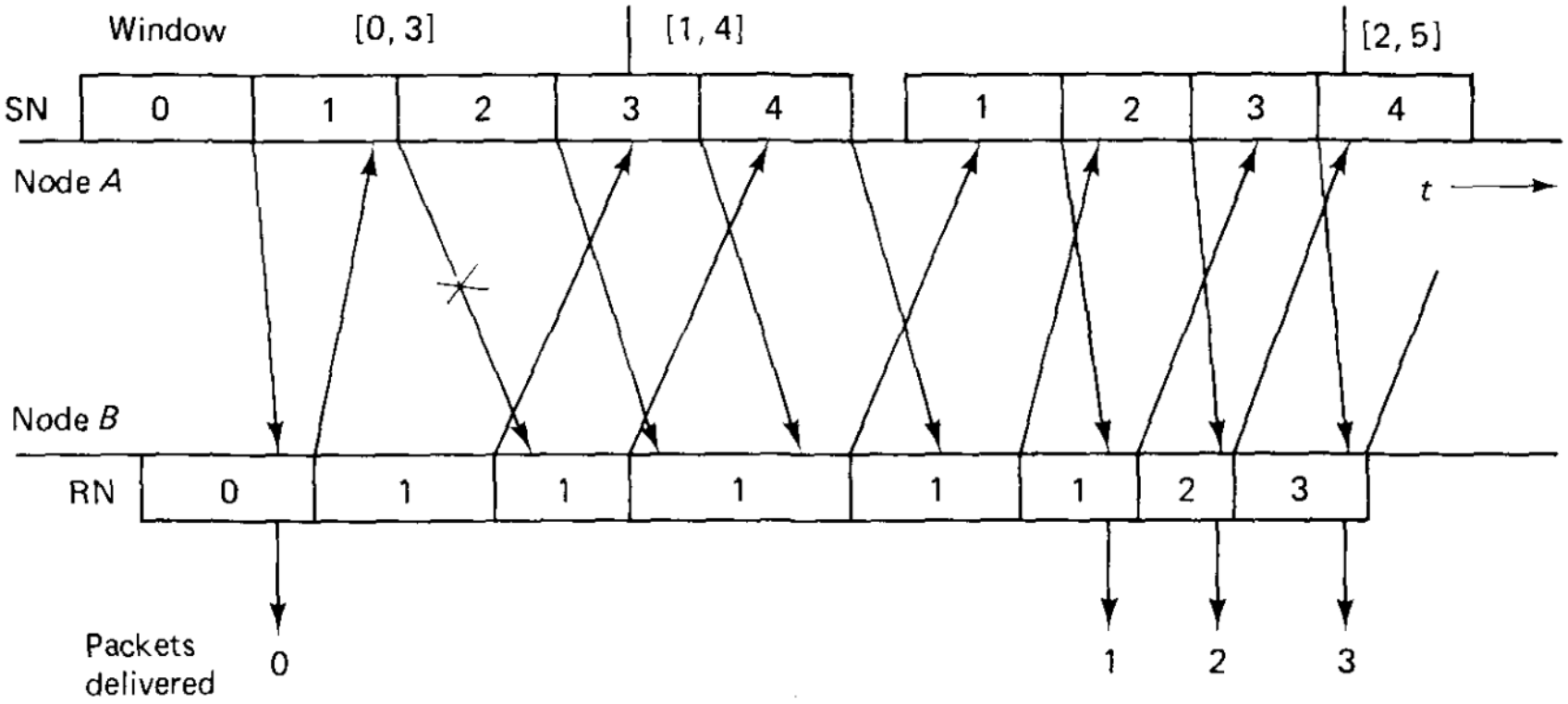
Suppose node A (transmitter) starts with SN=0 and node B (receiver) starts with RN=0.

What are the missing values?

Example of Go Back 7 ARQ

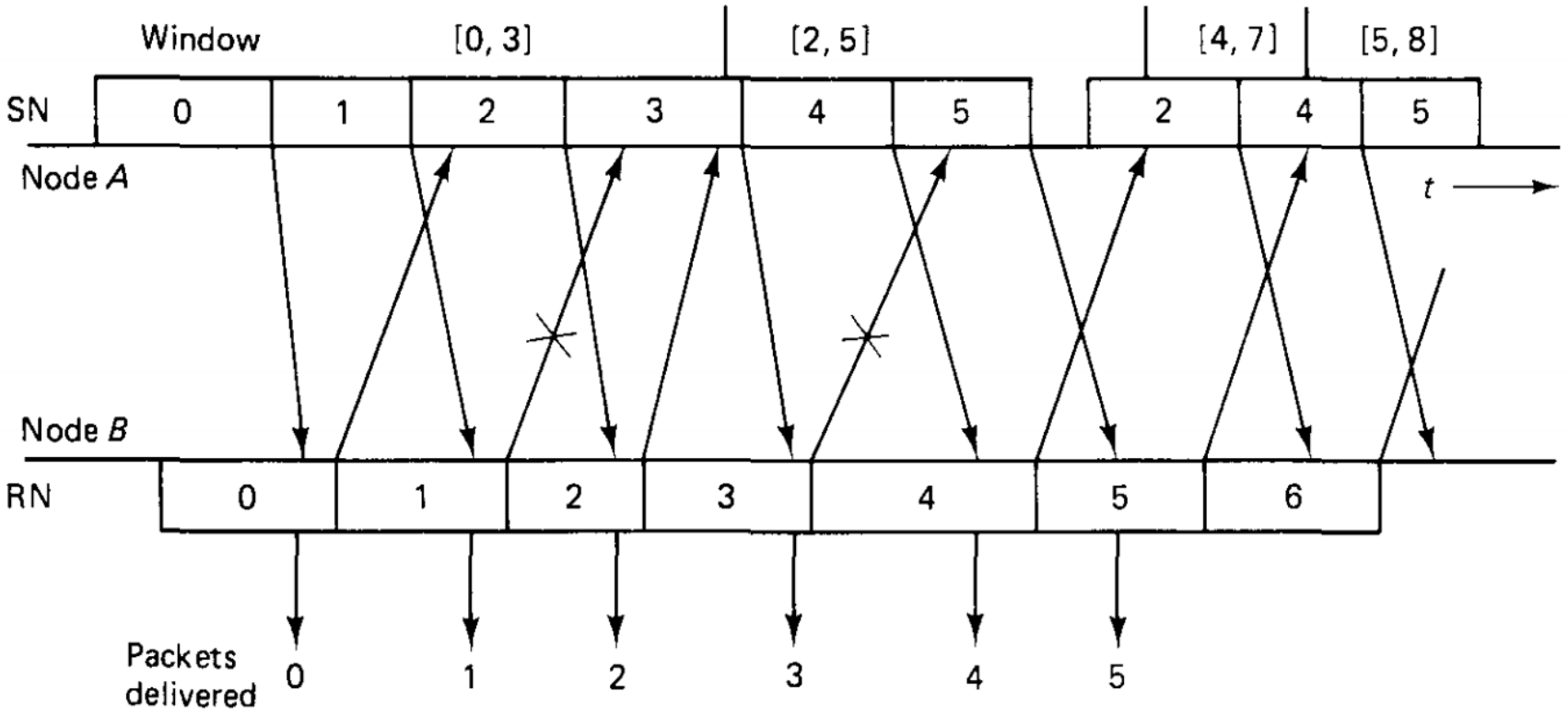


Example of Go Back 4 with Error



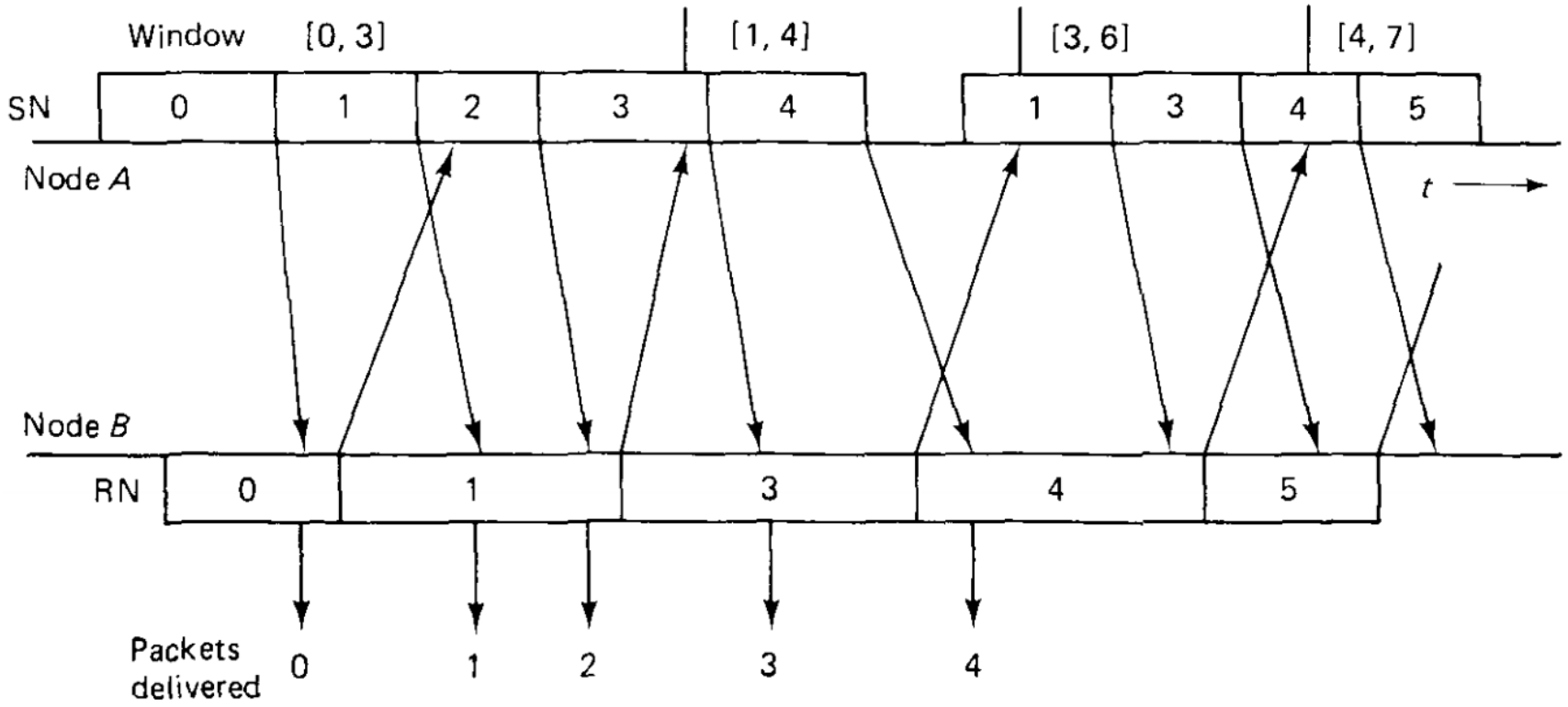
- **Entire window has to be retransmitted** after a forward error.
- Larger N, helps or makes it worse?

Example of Go Back 4 with (Reverse) Error



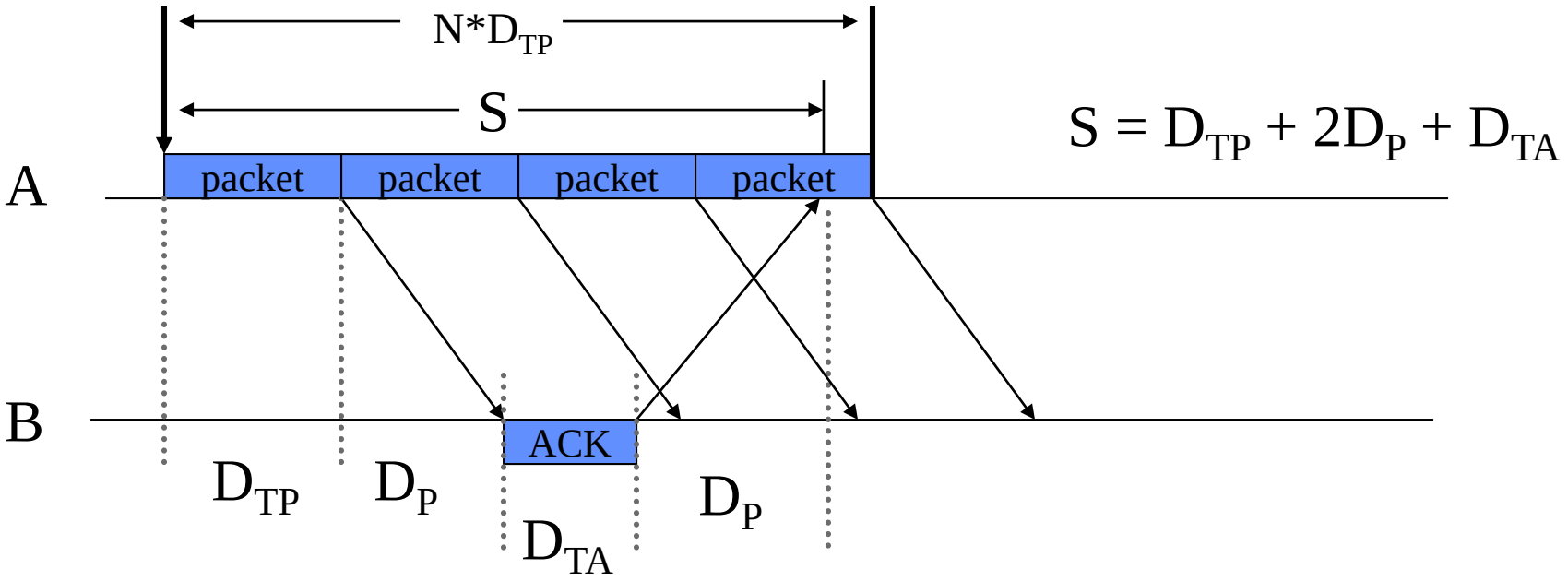
- When an error occurs in the reverse direction the ACK may still arrive in time.
- Packet 2 is retransmitted because RN = 4 did not arrive in time, however it did arrive in time to prevent retransmission of packet 3.
  - Larger N, helps?

Example of Go Back 4 with Long Frames



- Long frames in feedback direction slow down the ACKs.
- This can cause retransmissions.
- Larger N, helps?

Efficiency of Go Back N



- We want to choose N large enough to allow continuous transmission while waiting for an ACK for the first packet of the window,

$$N > S/ D_{TP}$$

- Without errors the efficiency of Go Back N is,

$$E = \min\{1, N*D_{TP}/S\}$$

Exercise

- **High-speed DSL Modem**
  - R = 10Mbps, Packet = 1000 bits, ACK = 1000 bits, D_p = 100μs
  - $D_{TP} = D_{TA} = D_p = 10^{-4}$ seconds ⇒ S = $4x10^{-4}$ seconds
  - Efficiency for N = 1?
  - Min N such that Efficiency = 100%?

- **Geo satellite link**
  - R = 1Mbps ⇒ $D_{TP} = D_{TA} = 10^{-3}$ seconds
  - D_p = 0.12 sec ⇒ S = 0.242 second
  - Efficiency for N = 1?
  - Min N such that Efficiency = 100%?

Efficiency of Go Back N with Errors

Assume: $N = \left\lceil \frac{S}{D_{TP}} \right\rceil$ TO = $N*D_{TP}$ P = prob. error

and when an error occurs, the entire window of N packets must be retransmitted.

Let X = the **number** of packets sent **per successful transmission.**

$E[X] = 1*(1-P) + (N+1)P(1-P) + (2N+1)P^2(1-P) + (3N+1)P^3(1-P) + \ldots$

$= (1-P)\,[\,(1 + P + P^2 + P^3 + \ldots) + NP\,(1 + 2P + 3P^2 + 4P^3 + \ldots)\,]$

$= (1-P)\,[\,1/(1-P) + NP/(1-P)^2\,] = 1 + N*P/(1-P)$

Efficiency = 1/E[X]

Go Back N Requirements

Go Back N is guaranteed to work **correctly**, independent of the detailed choice of which packets to repeat, if:

1) System is correctly initialized
2) No failures in detecting errors (CRC)
3) Packets travel in FCFS order
4) Positive probability of correct reception (1-P)
5) Transmitter occasionally resends SN_{min} (e.g., upon timeout)
6) Receiver occasionally sends RN

Correctness:

- Safe: algorithm never produces an incorrect result
- Live: it can continue forever (no deadlock)

Notes on Go Back N

- No buffering of packets at the receiver
- Packets can be numbered modulo M where M > N
  - Because at most N packets can be sent simultaneously
  - What would happen if we numbered packets modulo N?
- **The major problem with Go Back N is this need to re-send the entire window when an error occurs. This is due to the fact that the receiver can only accept packets in order.**
- **How can we solve this inefficiency?**

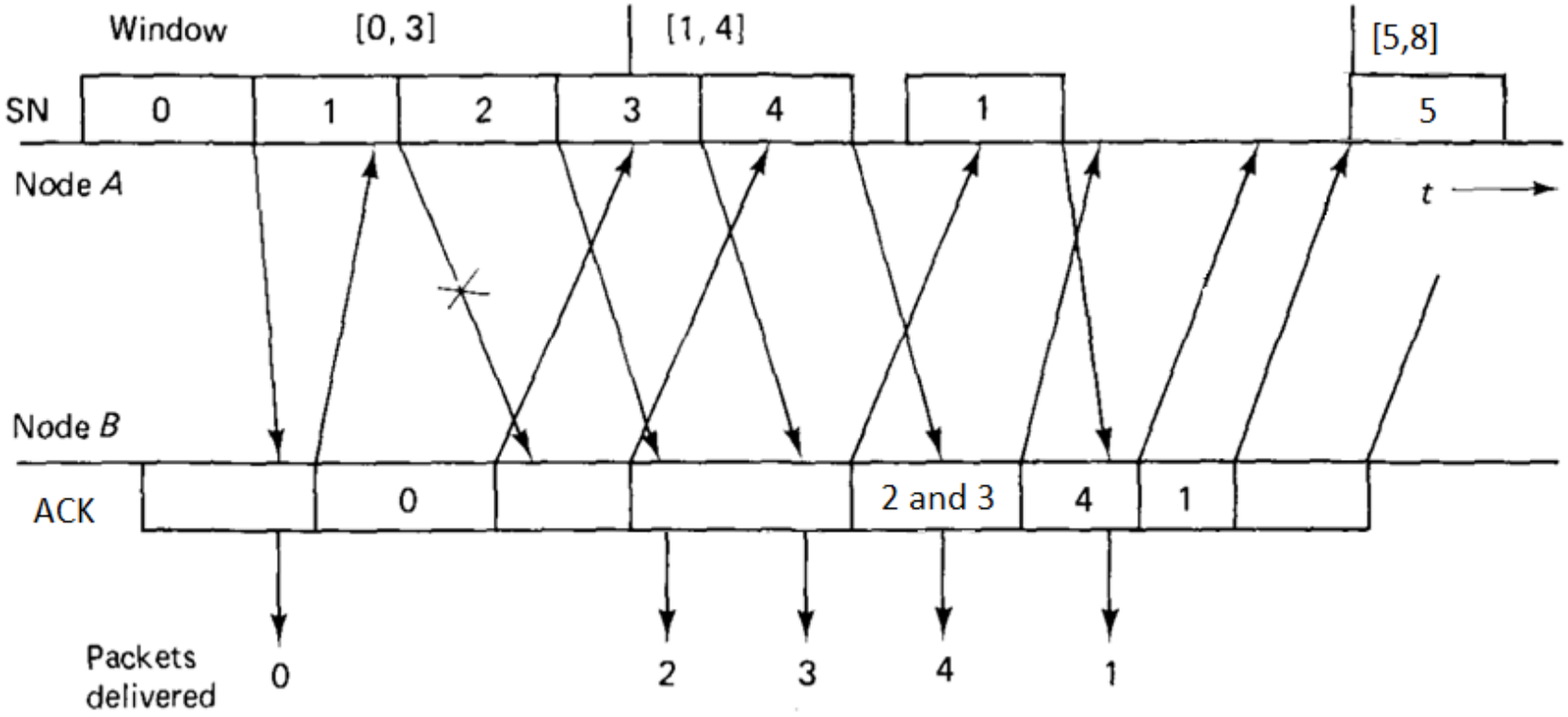
ARQ Protocols: Selective Repeat Protocol

Selective Repeat Protocol (SRP)

- Selective Repeat retransmits **only** packets that are lost due to errors
  - Receiver **accepts packets out of order** and ACKnowledges them individually.
  - Since receiver must **release packets to higher layer in order**, the receiver must be able to buffer some packets.

- SRP rules for window size W:
  - Sender is the same as in Go Back N:
    - Transmits new packet if it is within W of ALL un-ACKed packets.
    - Retransmit un-ACKed packets after a timeout
  - Receiver ACKs **each and every** correct packet
  - Packets are numbered mod M where $M \geq 2W$

Example of SRP (W = 4) with Error



- **Compare with Go Back N**

Efficiency of SRP

- For **ideal SRP**, only packets containing errors will be retransmitted
  - Ideal is not realistic because sometimes packets may have to be retransmitted because their window expired. However, if the window size is set to be much larger than the timeout value then this is unlikely
- Assuming ideal SRP, efficiency = 1 - P

  P = probability of a packet error
- Notice the difference with Go Back N where

$$\text{efficiency (Go Back N)} = 1/(1 + N*P/(1-P))$$

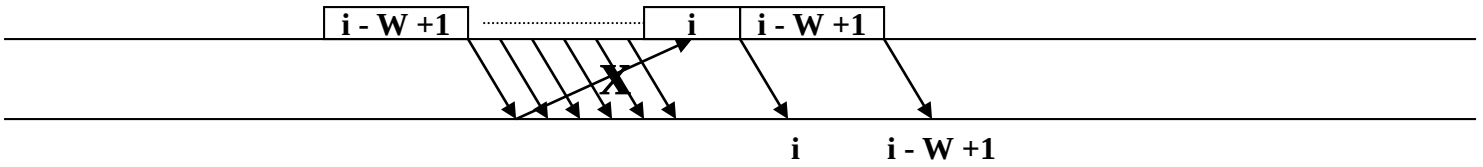
- When the window size is small performance is about the same, however with a large window, SRP is much better
  - As transmission rates increase we need larger windows and hence the increased use of SRP

Efficiency: Go Back N vs. SRP

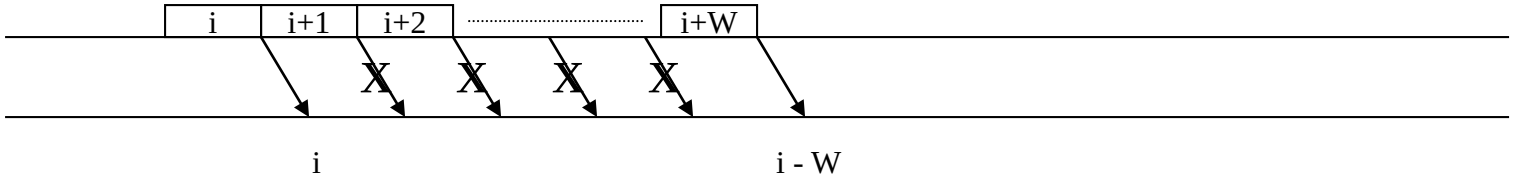
- Consider a satellite link with:
  - N = 1000 packets
  - p = 0.01 (1% of packets in error)
- SRP efficiency = 1 – p = 99%
  - Note that this is true for all N
- Go Back N efficiency = 1/(1 + N*P/(1-P)) = 9%
  - Note that this would get worse as N gets larger

Why are packets numbered Modulo 2W?

- Lets consider the range of packets that may follow packet i at the receiver



Packet i may be followed by the first packet of the window (i -W+1) if it requires retransmission



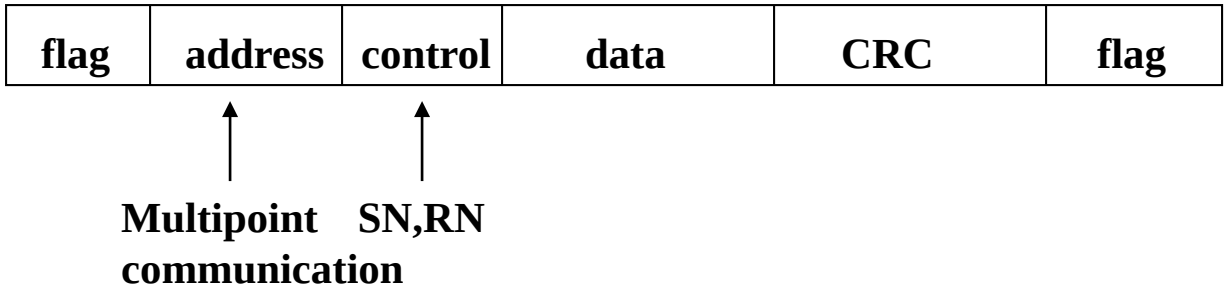
Packet i may be followed by the last packet of the window (i+W) if all Of the ACKs between i and i +W are lost

- Receiver must differentiate between packets i -W+1 ... i +W
  - These 2W packets can be differentiated using Mod 2W numbering

Standard DLCs

- HDLC, PPP, LAPB (X.25), and SDLC are very similar
  - HDLC/ SDLC developed by IBM for IBM SNA networks
  - PPP is commonly used for providing framing, error detection
- They all use bit oriented framing with flag = 01111110
- They all use a 16-bit CRC for error detection
- They all use Go Back N ARQ with N = 7 or 127 (optional)

SDLC packet



- Older protocols (used for modems, e.g., xmodem) used stop and wait and simple checksums

Reliability in the Protocol Stack