

# **CMPE 150: Introduction to Computer Networks**

**Dr. Chane L. Fullmer**  
**chane@cse.ucsc.edu**

# Mid Term

- Next Tuesday – April 29<sup>th</sup>
  - **Bring Scantron (large pink one)**
  - 30-50 questions
    - Multiple choice
    - Questions come from both text and lectures
      - Some may be similar to homework
  - ***Closed book, closed notes***

# Homework Assignments

Homework assignment #2

Due by May 1.....

# **CMPE 150: Introduction to Computer Networks**

## **LECTURE 7:**

### ***Logical Link Control, Part I***

# Logical Link Control

- MAC protocol provides best effort service.
- Even when ACKs are used in the MAC, the LLC layer can decide when to retransmit.
- LLC bridges the gap between service expected by network layer and service provided by MAC layer.
- LLC uses the header information in MAC frames.
- Example: PPP (LLC) over Ethernet (MAC)

# Functions at The Link Layer

- MAC:
  - Framing
  - Error checking
  - Naming within LANs (with MAC addresses)
  - Sharing of medium
  - Flow control (in some cases)
- LLC:
  - **Retransmission strategy**
  - **Link management**  
(deciding when a link exists or does not)
  - Flow control (in some cases)

# Types of Service LLC Can Provide

- ***Unacknowledged connectionless service:*** Datagram transmission, no connection exists, no error checking, *framing is the only service provided (e.g., SLIP).*
- ***Acknowledged connectionless service:*** No connections, each frame is ACKed individually. This service can be provided as part of the MAC itself (e.g., CSMA/CA protocols)
- ***Connection-Oriented Service:*** Data exchanged within a connection, provides net layer with a virtual reliable packet stream (e.g., HDLC, PPP)

# Retransmission Strategies

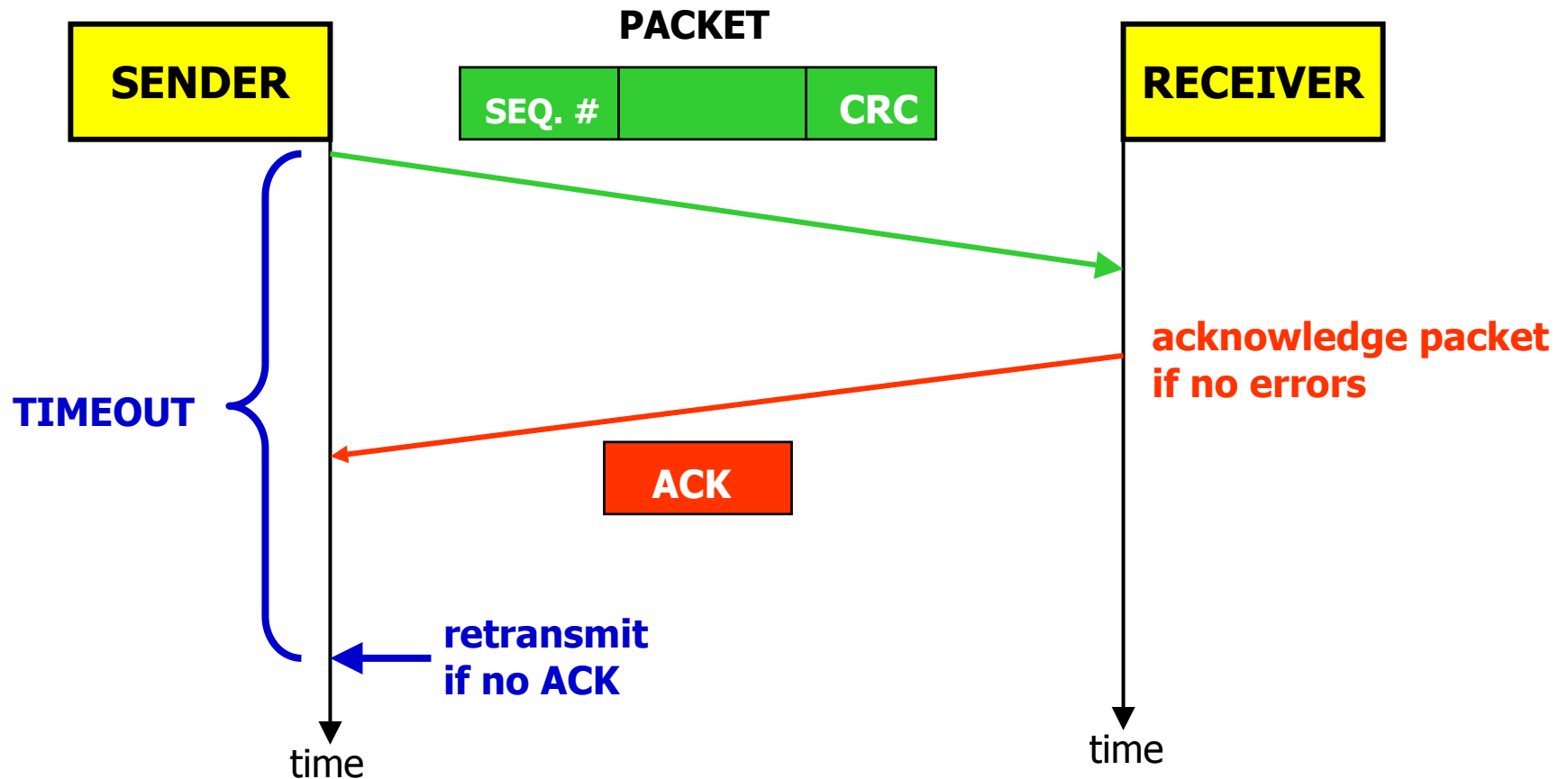
- Called Automatic Repeat reQuest (ARQ) algorithms or protocols.
- Read Sections 3.4 of text.
- Design Issues:
  - How does a receiver ask for a retransmission?
  - How does the sender know when to retransmit?
  - How are packets associated with requests for retransmissions?
- Performance and *correctness*?
- We will assume point-to-point links for simplicity.

# ARQ Assumptions

- The protocol operates at the link layer.
- One sender and one receiver.
- Links and nodes behave in FIFO order.
- Nodes execute ARQ protocol correctly.
- Session between sender and receiver is already initialized and is permanent.
- All errors are detected correctly and framing is done perfectly.

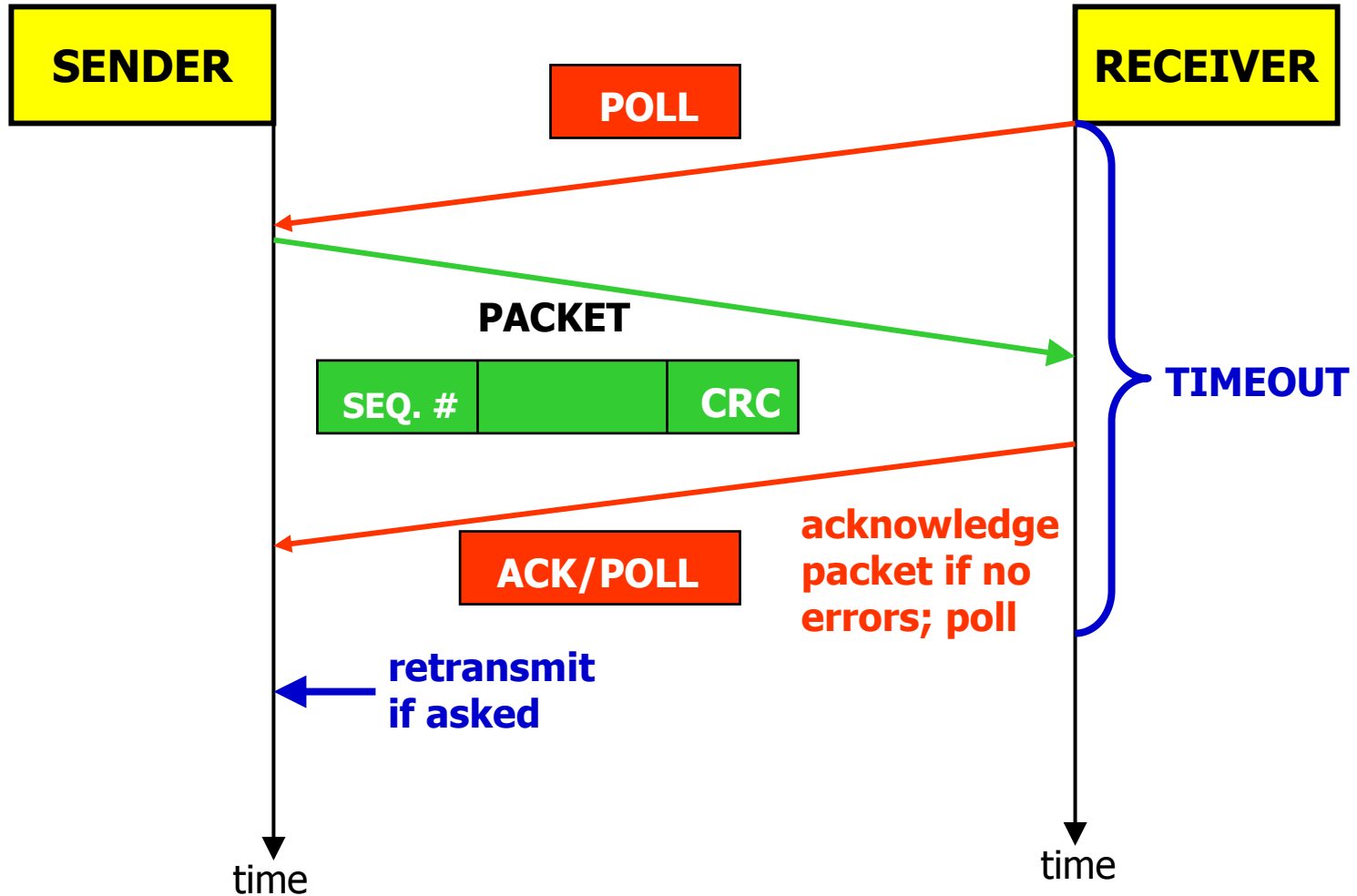
# Generic ARQ Scheme

## SENDER INITIATED



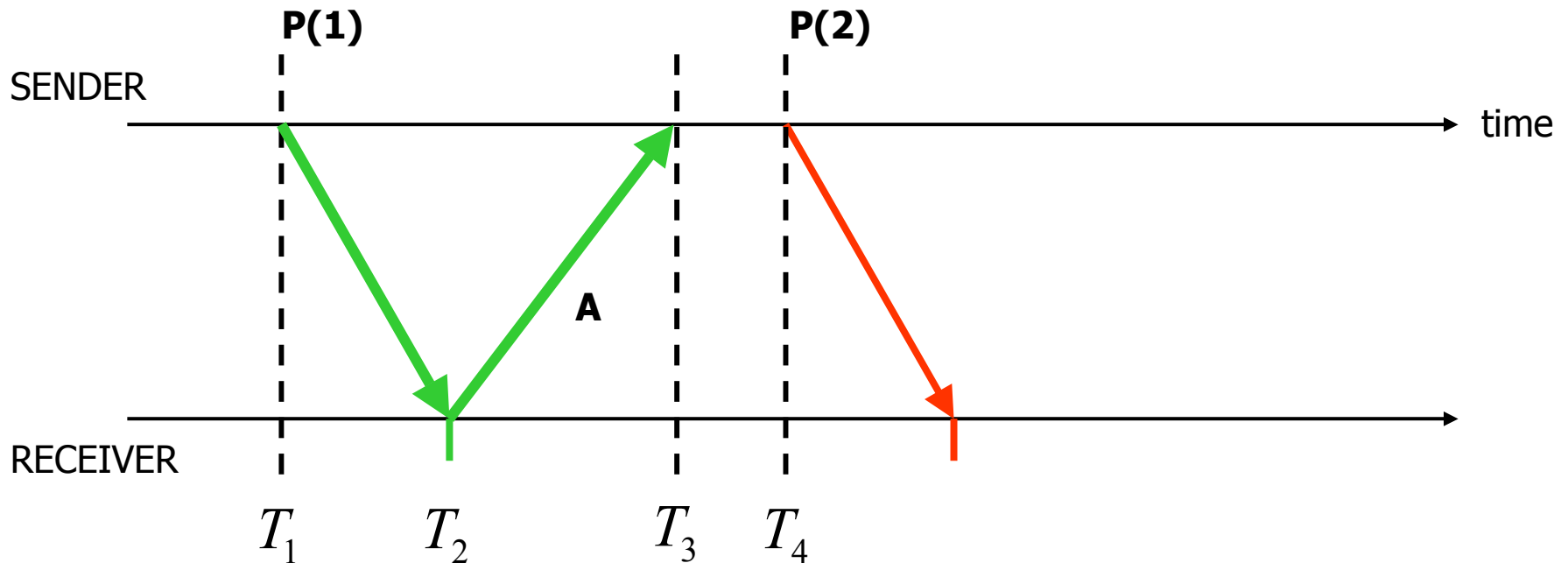
# Generic ARQ Scheme

## RECEIVER INITIATED



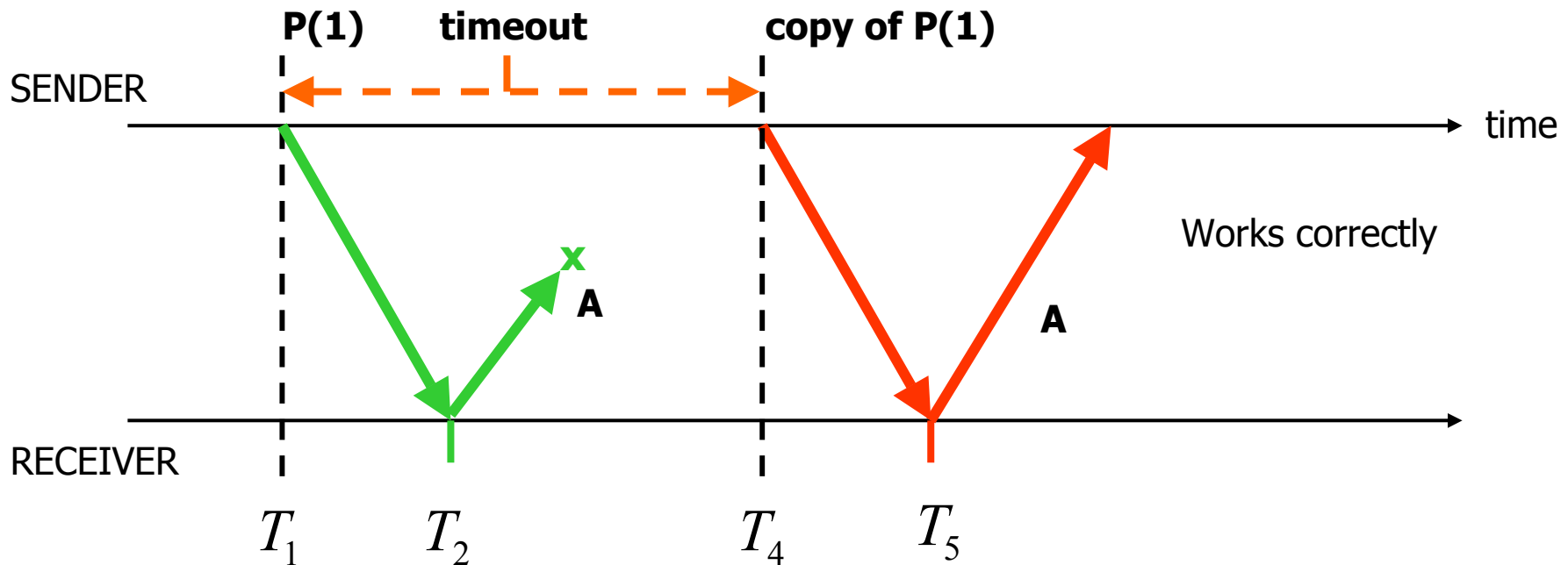
# Example of Simple ARQ

- Assume receiver labels every ACK with the same label (A), regardless of the packet being acknowledged.

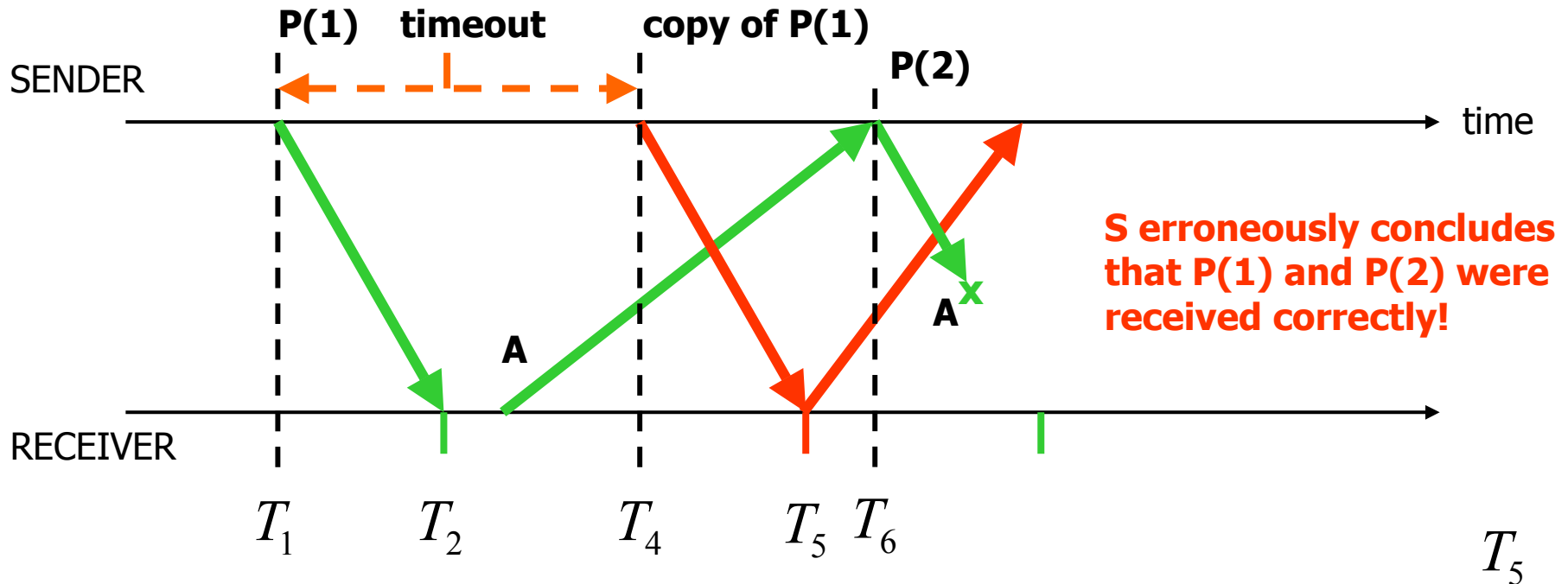


# Errors Can Confuse Receiver

- Assume that sender retransmits packet when it receives no ACK within a timeout.



# Errors Can Confuse Receiver



To avoid confusion at S, each ACK must reference the packet for which it is sent.

# Requirements in ARQ

- Sender labels each packet it sends using a linear sequence-number space.
- Receiver ACKs each packet it receives without errors and numbers each ACK with the sequence number of the corresponding packet.
- Sender times out after not receiving an ACK for the packet within some finite amount of time, and retransmits the packet then.
- Sender sends up to a certain number of un-ACKed packets.

# Stop-and-Wait ARQ

- Sender transmits packets labeled 1, 2,.....
- Receiver ACKs *every packet received correctly* and ACK specified the packet being acknowledged (or next expected packet).
- Receiver passes copy of packet correctly received to the network layer and drops packets with errors.
- Sender retransmits copy of packet if no ACK arrives within a timeout interval.
- Sender and receiver are initialized to start sending and receiving packet 1.

# Stop-and-Wait Protocol (SWP)

## Sender's algorithm:

1. Initialize packet number (PN) to 0 [or 1 for example].
2. Wait until packet is received from network layer; assign PN to packet.
3. Transmit (or retransmit) packet in frame with PN as its sequence number.
4. Start timeout timer and wait for ACK from receiver.
5. If error-free ACK specifying PN is received within timeout interval,  
    then set  $PN = PN + 1$  and go to Step 2.  
    else go to Step 3

# Stop-and-Wait Protocol

Receiver's algorithm:

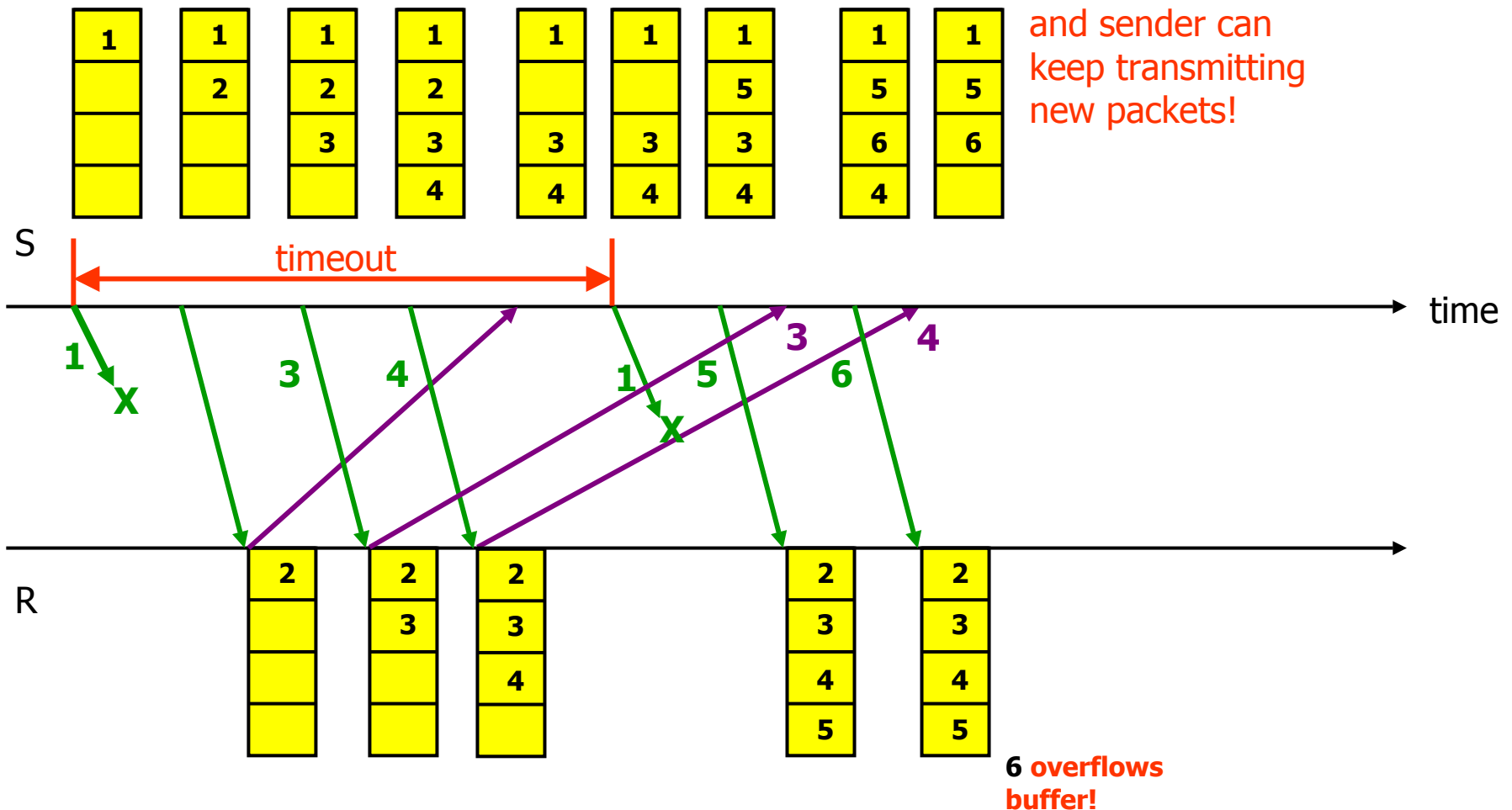
1. Initialize next packet (NP) to 0 [or 1 for example] and repeat Step 2 forever
2. If ( packet is received )  
then  
if (error in packet)  
then discard packet  
else do  
if ( PN=NP )  
then give packet to network layer and set NP = NP+1  
send ACK containing PN (or NP)

# Selective Repeat ARQ

- **Motivation:** SWP leaves sender idle for long periods of time waiting for ACKs.
- **Solution:** Allow sender to transmit multiple packets while waiting for the ACK of a given packet. **Have a pipeline of packets!**
- **Requirements:**
  - Sender and receiver can buffer a number ( $W$ ) of packets
  - Sender labels packets using consecutive numbers 1, 2, ....
  - Receiver buffers packets received without error, ACKs them, and delivers packets to network layer in the *correct* order  
(e.g., if packet P1 is in error and P2 and P3 are received correctly, the receiver buffers them until it receives P1 correctly)
  - Sender buffers copies of transmitted packet until it receives the corresponding ACK.
  - Sender retransmits a packet when its timeout expires with no ACK.
  - ACK refers to the sequence number of the packet it acknowledges.

# How Many Packets Allowed in Sender's Buffer?

Assume we allow a maximum of  $W=4$  in buffer





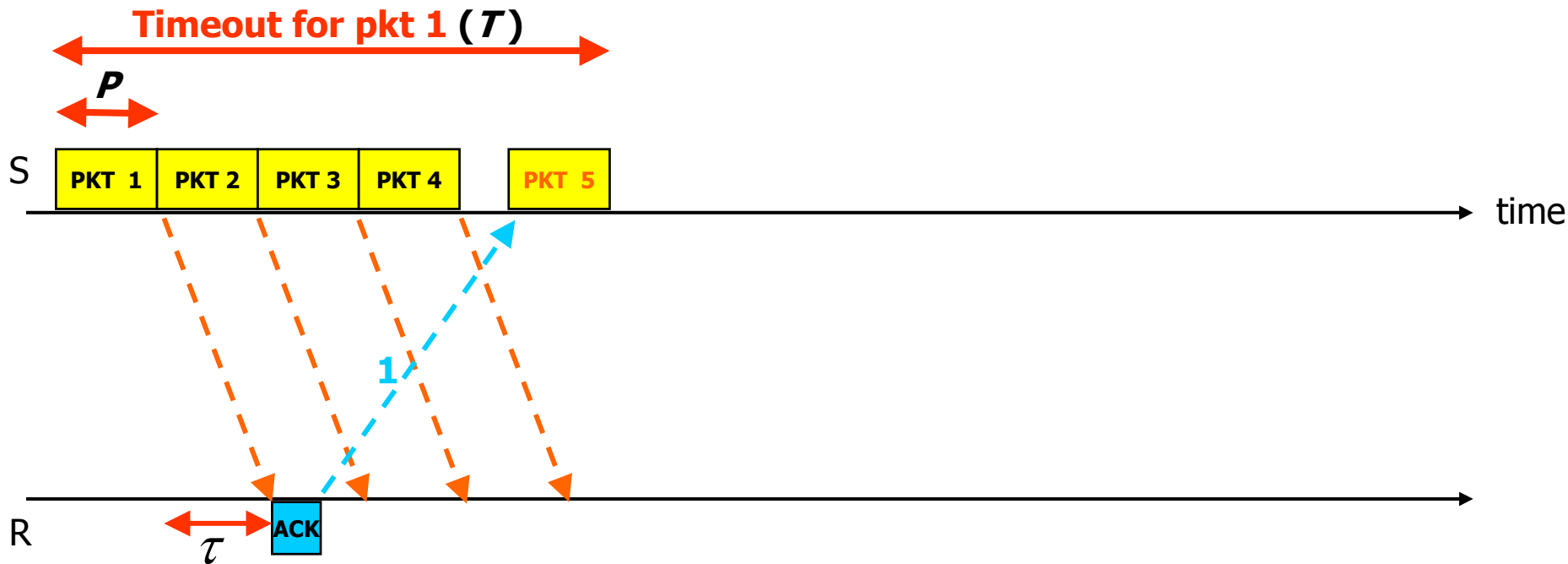
# Selective Repeat Protocol

- Write a pseudocode description of SRP!
- Consider sender and receiver.

# Go-Back-N (GBN) ARQ

- SRP requires sender and receiver to have a buffer, which is not an issue today.
- With GBN, the receiver discards any packet it receives out of order; therefore, it does not need a buffer.
- Receiver accepts only those packets received in order.
- Receiver ACKs a packet received correctly with the sequence number of the *last packet received in sequence*.
- The sender starts a timer for each packet it transmits, and after the timeout of a packet expires, *it retransmits the packet and all the packets sent after that packet*.
- Sender can have up to  $W$  packets waiting for ACKs.

# GBN ARQ Example 1

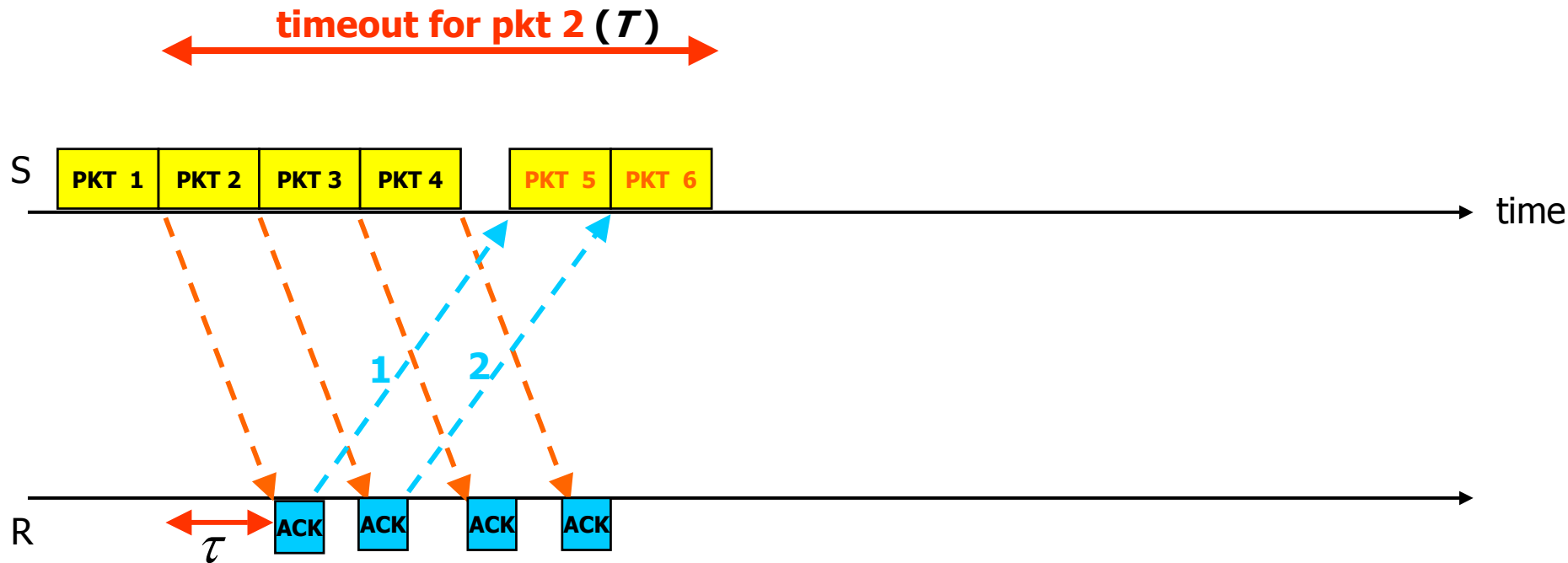


**Packet 1 is received correctly; receiver sends ACK**

**Sender is able to transmit four packets before timeout for pkt 1 expires**

**ACK to pkt 1 arrives before its timeout expires; sender can send packet 5.**

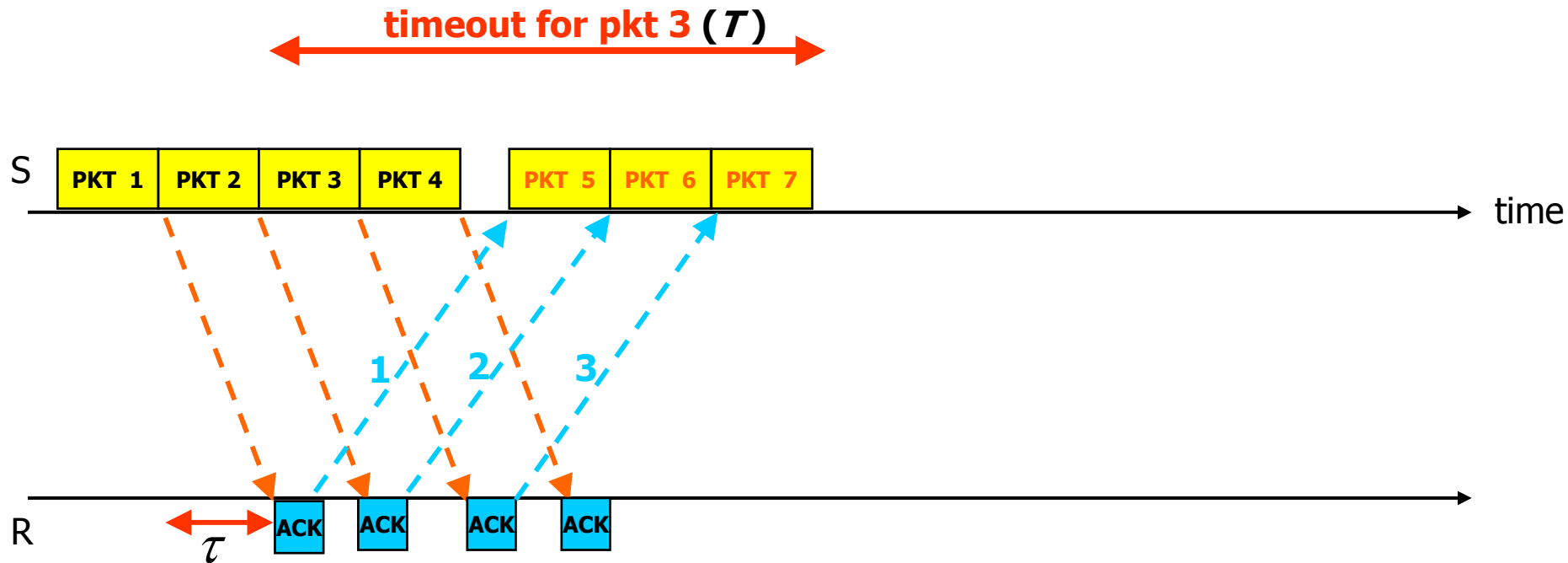
# GBN ARQ Example 1



**Packets 2 to 4  
arrive in order at  
the receiver;  
receiver sends ACKs  
for them**

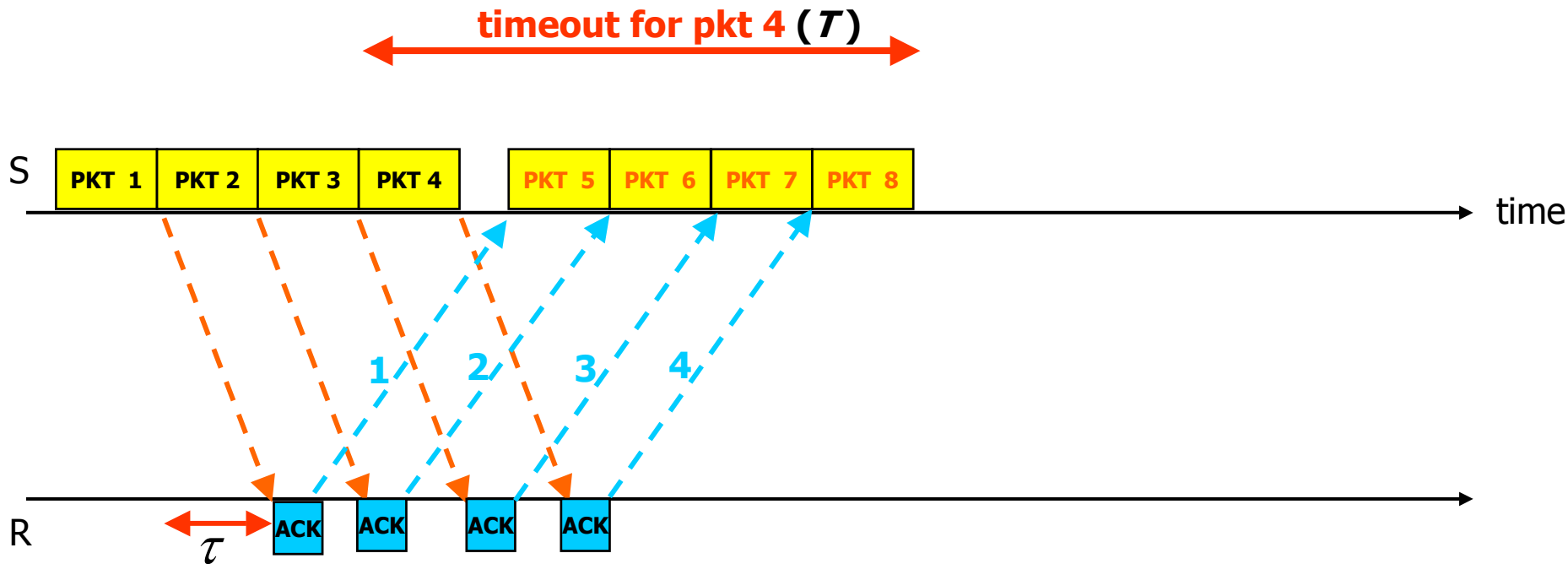
**ACK to pkt 2 arrives  
before its timeout  
expires; sender can  
send packet 6.**

# GBN ARQ Example 1



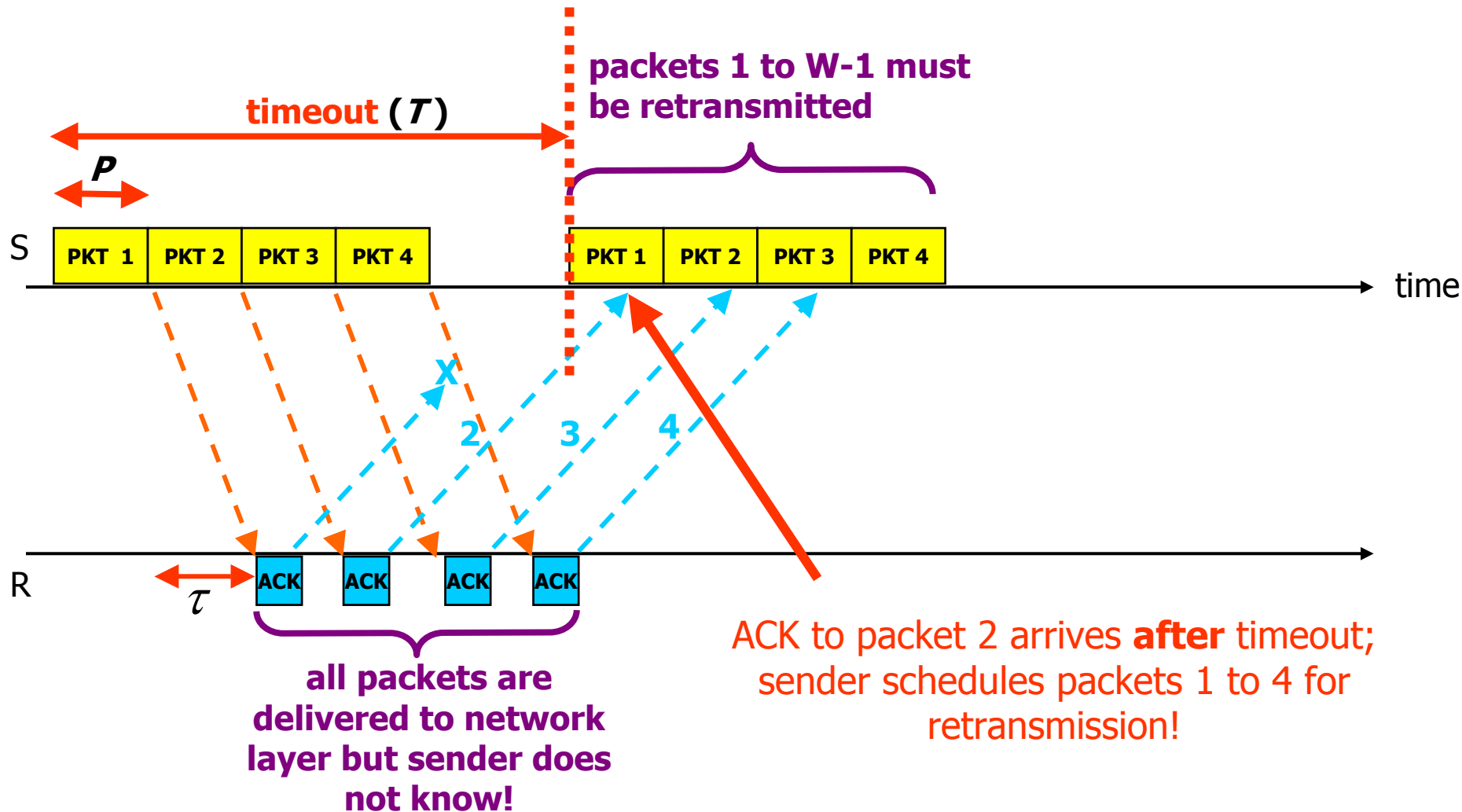
**ACK to pkt 3 arrives before its timeout expires; sender can send packet 7.**

# GBN ARQ Example 1

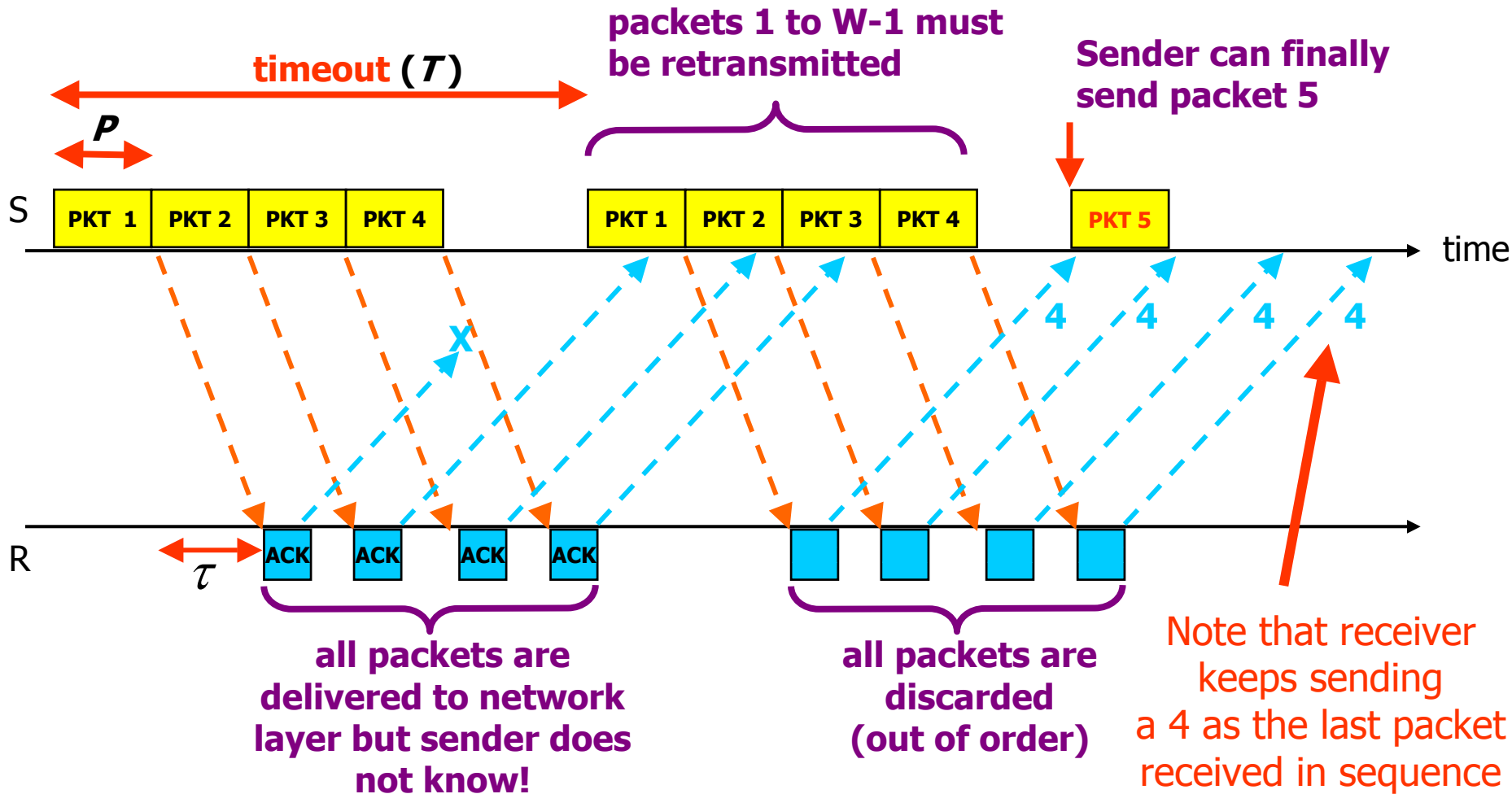


**ACK to pkt4 arrives before its timeout expires; sender can send packet 8, etc.**

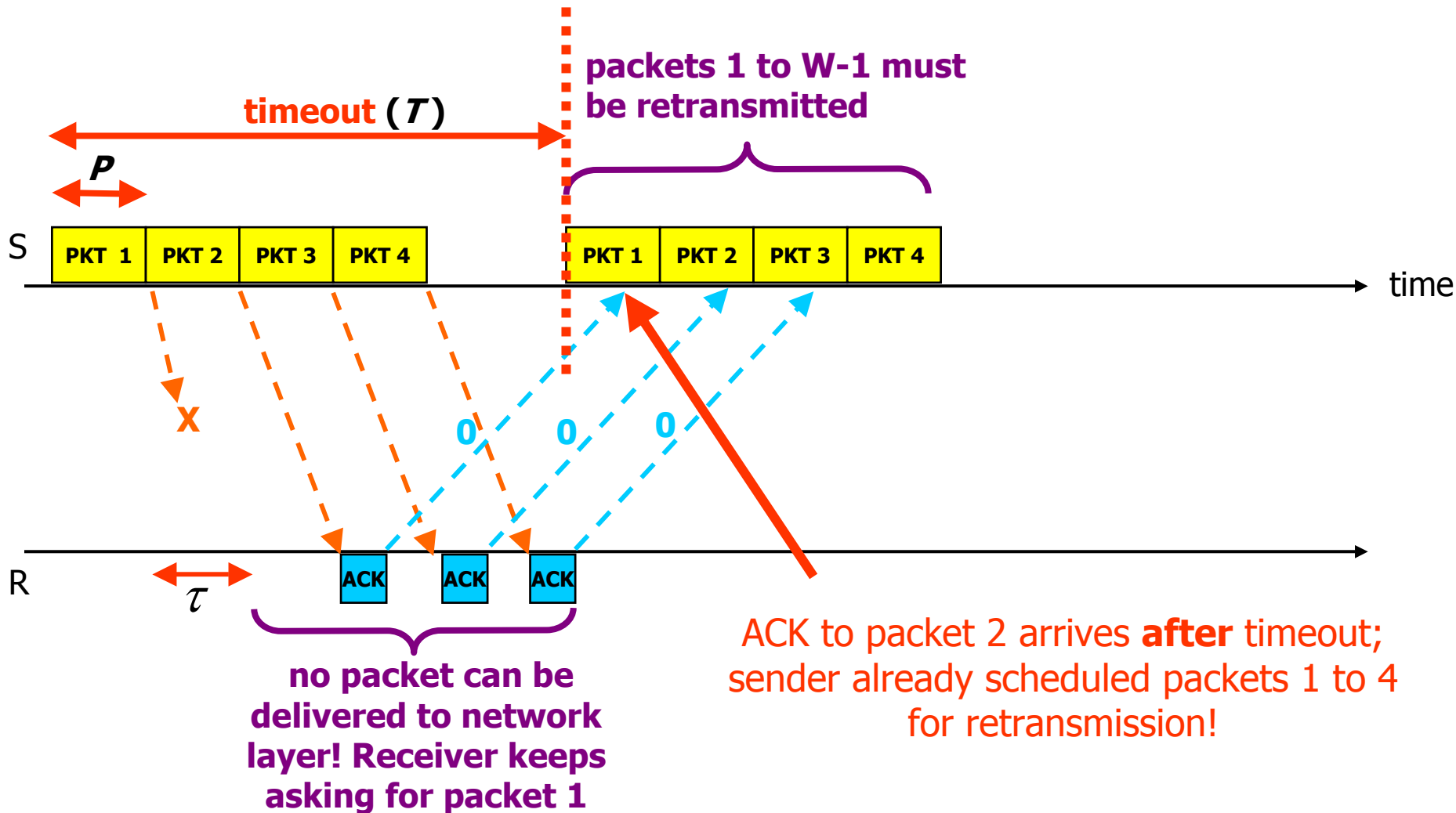
# GBN ARQ Example 2



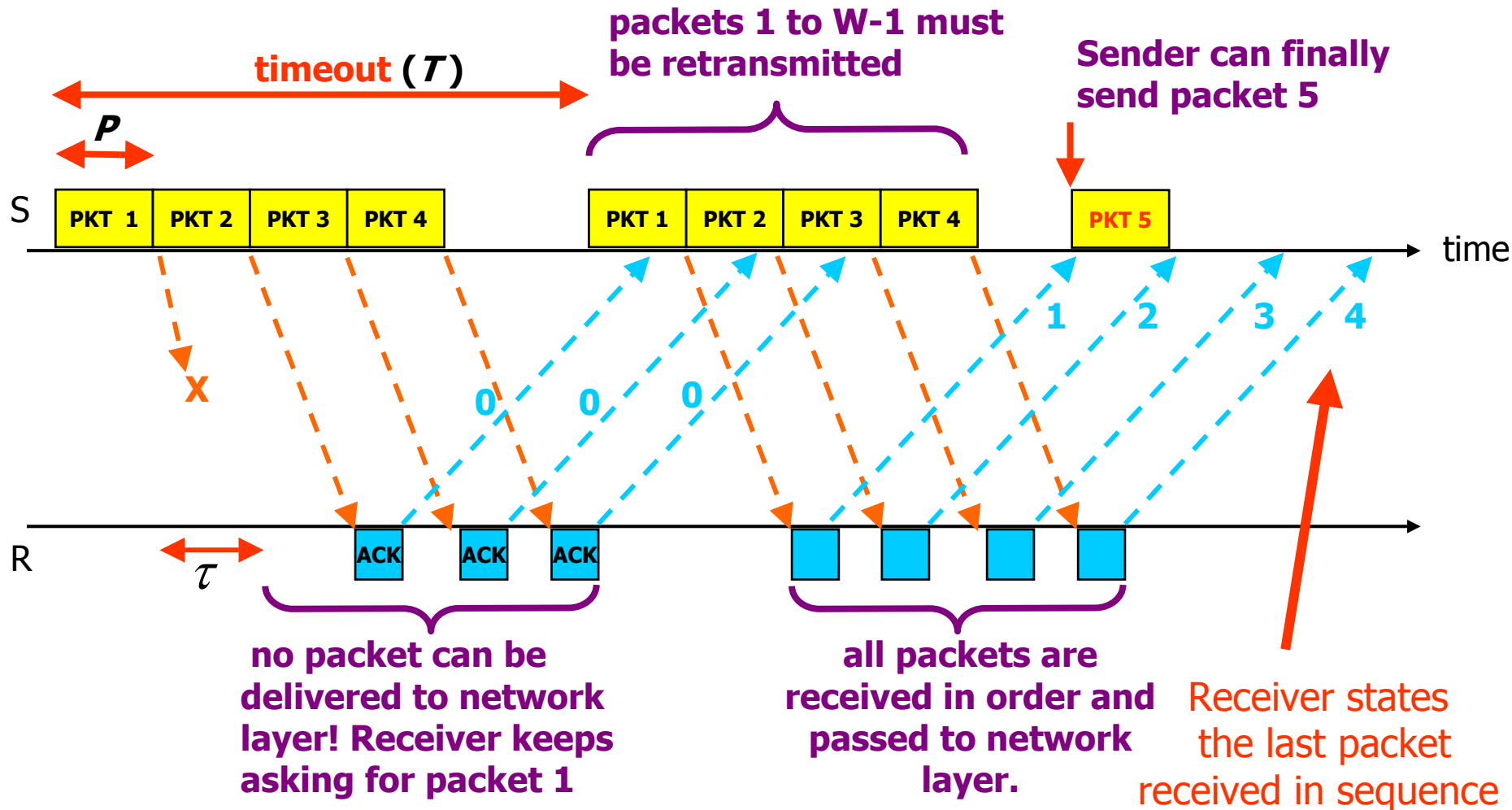
# GBN ARQ Example 2



# GBN ARQ Example 3



# GBN ARQ Example 3

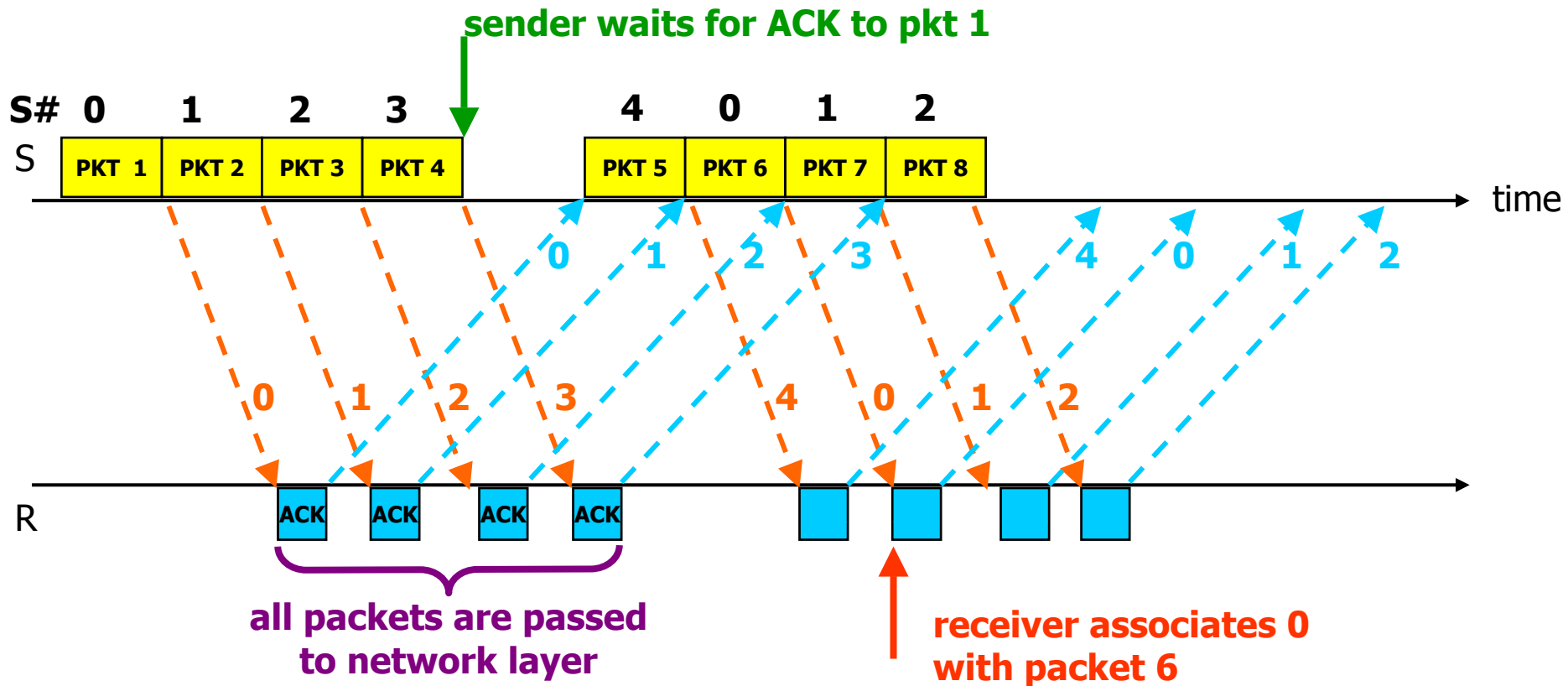


# Correctness of SRP

- What to prove:
  - ***Finite sequence number space:*** What is the minimum sequence number space we can use so that sender and receiver never get confused with sequence numbers in ACKs and packets.
  - ***Safety:*** Receiver passes packets to the network layer in sequence, without gaps or replicas.
  - ***Liveness:*** Deadlocks never occur.

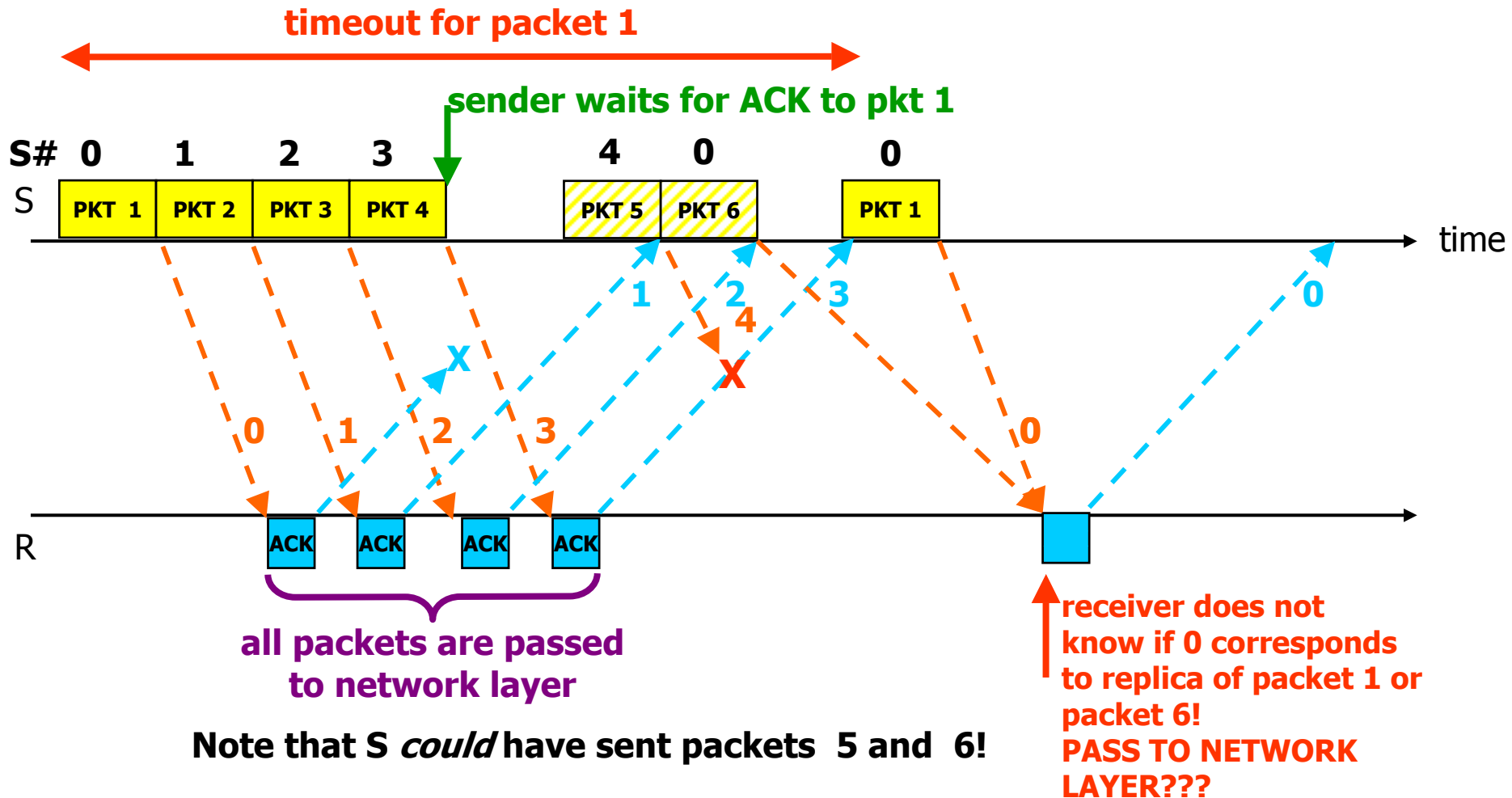
# Sequence Numbering in SRP

- Assume  $W=4$  and sequence numbers are seven sequence numbers (i.e., 0 to 4)



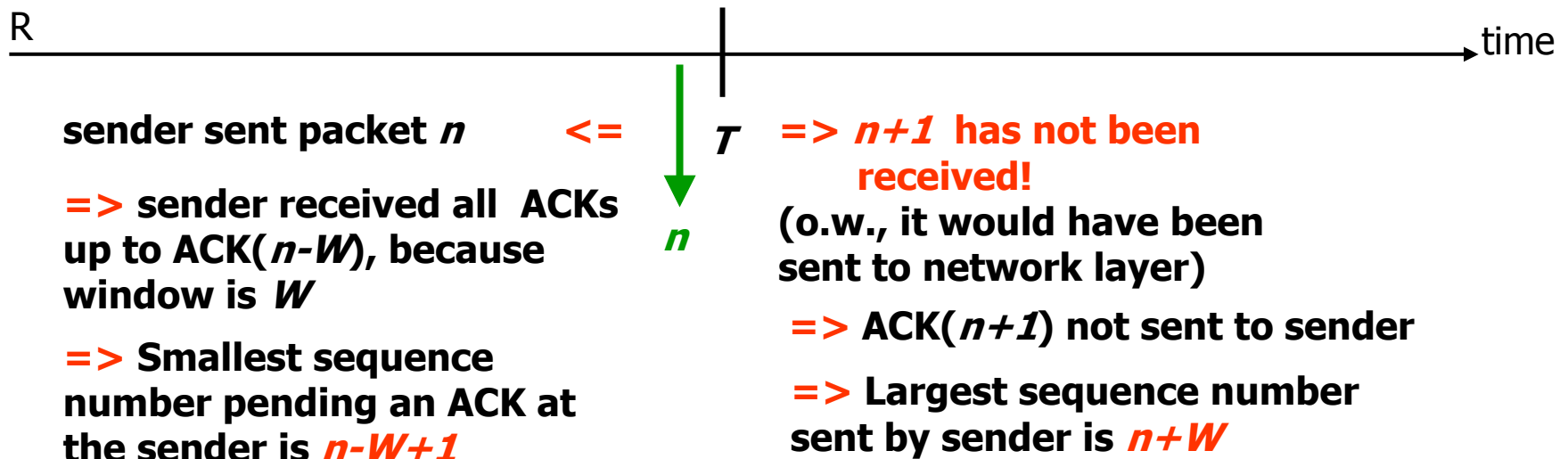
# Sequence Numbering in SRP

- However, receiver can get confused!



# Sequence Numbering in SRP

- Assume that window is  $W$  and packets are numbered modulo  $2W$  (**from 0 to  $2W-1$** )
- Assume that, at time  $T$ , packet labeled  $n$  is passed to network layer at the receiver (and is the packet with the highest number that can be passed to net layer).



Given  $n$ , the possible range of sequence numbers of packets at the sender is  $\{n-W+1, n+W\}$  and using modulo  $2W$  sequence number space works correctly

# Sequence Numbering in SRP

- The same type of argument can be made for the receiver, assuming that the sender receives the ACK for packet  $n$  for the first time at time  $T$ .
- Left to the student to show that using a sequence number space modulo  $2W$  does not confuse the receiver!
- Show the same for SWP!
  - $W = 1$

# Safety Proof for SRP

- **By induction on sequence number  $PN$  of a packet.**
- **Prove for the first packet:**
  - S and R are initialized with  $PN = NP = 0$ .
  - First packet sent by S is 0 and S can only send up to  $PN = W - 1$  to R due to window size.
  - S must retransmit packet with  $PN = 0$ , until it receives an ACK with  $NP > 0$  or  $NP = 0$  from R
  - SEE WHY?
  - *Therefore, SRP is safe for packet 0!*

# Safety Proof for SRP

- **Assume that SRP is safe for  $PN = n-1$  and prove for  $PN = n$ :**
  - R has set  $NP = n$
  - R can only accept a packet with  $PN = NP$  to be passed to the network layer, which can be done together with a set of other packets whose sequence numbers are in sequence.
  - When the first error-free copy of packet with  $PN = n$  arrives, R passes it to network layer in sequence and increases  $NP$ . No replica of the same packet is given to the network layer and no missing packets exist from 0 to  $n$ ; therefore, SRP is safe.

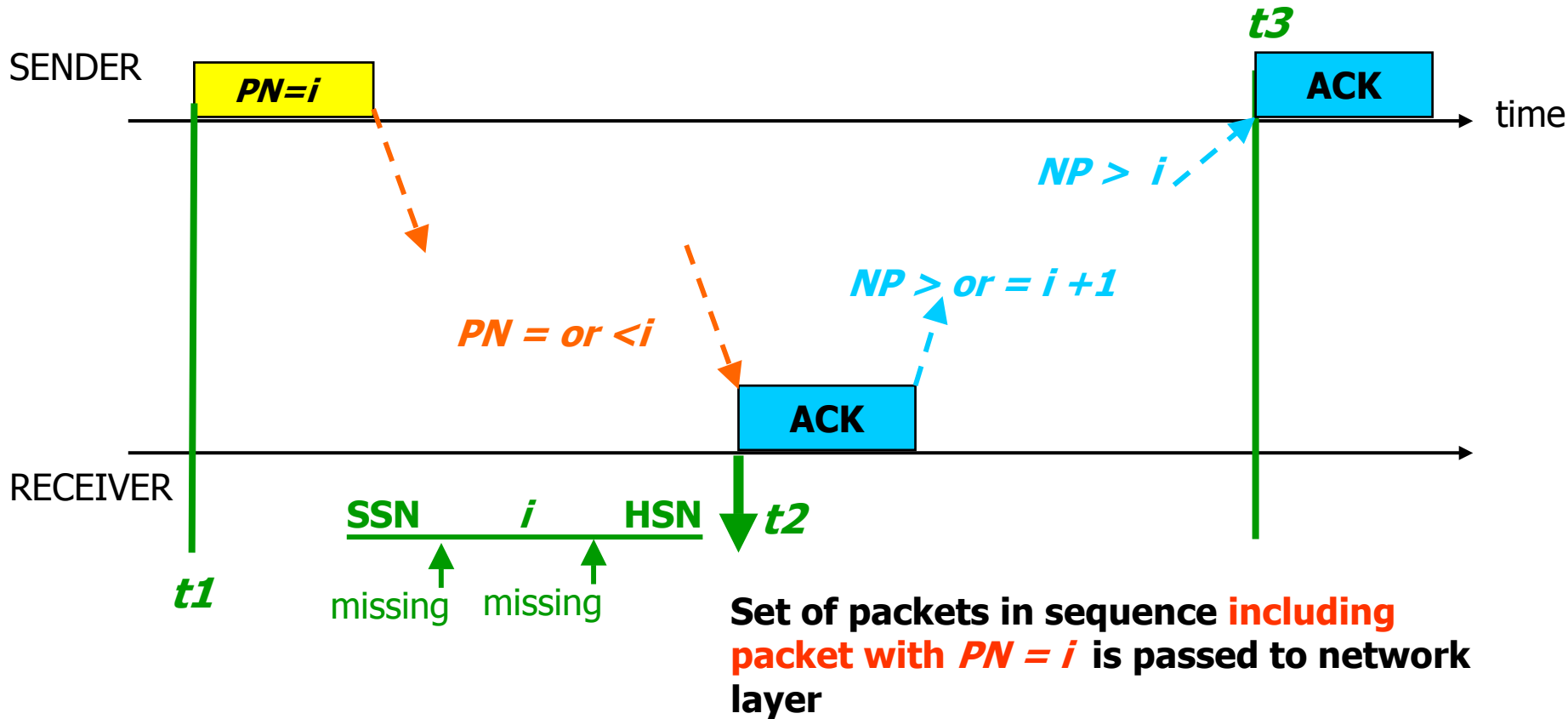
□

Q.E.D.

# Liveness Proof for SRP

- The trick is to identify key events in time ensuring that the protocol never deadlocks.
- In an ARQ protocol, the key events can be stated with respect to a given packet  $n$ :
  - The **first transmission from S** of packet with  $PN = n$  occurs at time  $t1$
  - The first time that **R receives** an error-free transmission of a **packet with any  $PN < or = n$**  that makes the packet with  $PN = n$  be in sequence occurs at  $t2$
  - **S receives** an error-free transmission of an **ACK** reporting  $NP = or > n+1$  at  $t3$

# Liveness Proof of SRP



NO DEADLOCKS IF  $[t_1, t_3] < \infty$

# Liveness of SRP

By definition of SRP,  $NP(t)$  and  $PN(t)$  are non-decreasing functions of  $t$

Because the channel has a propagation delay and  $t_1$  is the first time when S releases a packet with  $PN = i$ , given that SRP is safe, it is impossible for R to release a packet with  $PN = i$  before or at  $t_1$ ; therefore,  $t_2 > t_1$

Because the probability that any packet and its ACK goes through the channel correctly is  $> 0$ , it follows that  $[t_1, t_2]$  is finite.

Because R releases the first ACK with  $NP \geq i$  at time  $t_2$  and because the channel has a positive propagation delay, it must be true that  $t_3 > t_2$

Because the probability of receiving an ACK with an NP covering a packet with  $PN < NP$  is non-zero, it follows that  $[t_2, t_3]$  is finite.

*Q.E.D.*

# Effect of Negative ACKs

- We have assumed that packets are retransmitted after a timeout.
- Using negative acknowledgments (NACKs), the receiver tells the sender about missing packets and the sender can retransmit after receiving the NACK. This increases throughput.
- NACKs and ACKs can be combined in windows of ACKs:



That's all

for Today!!

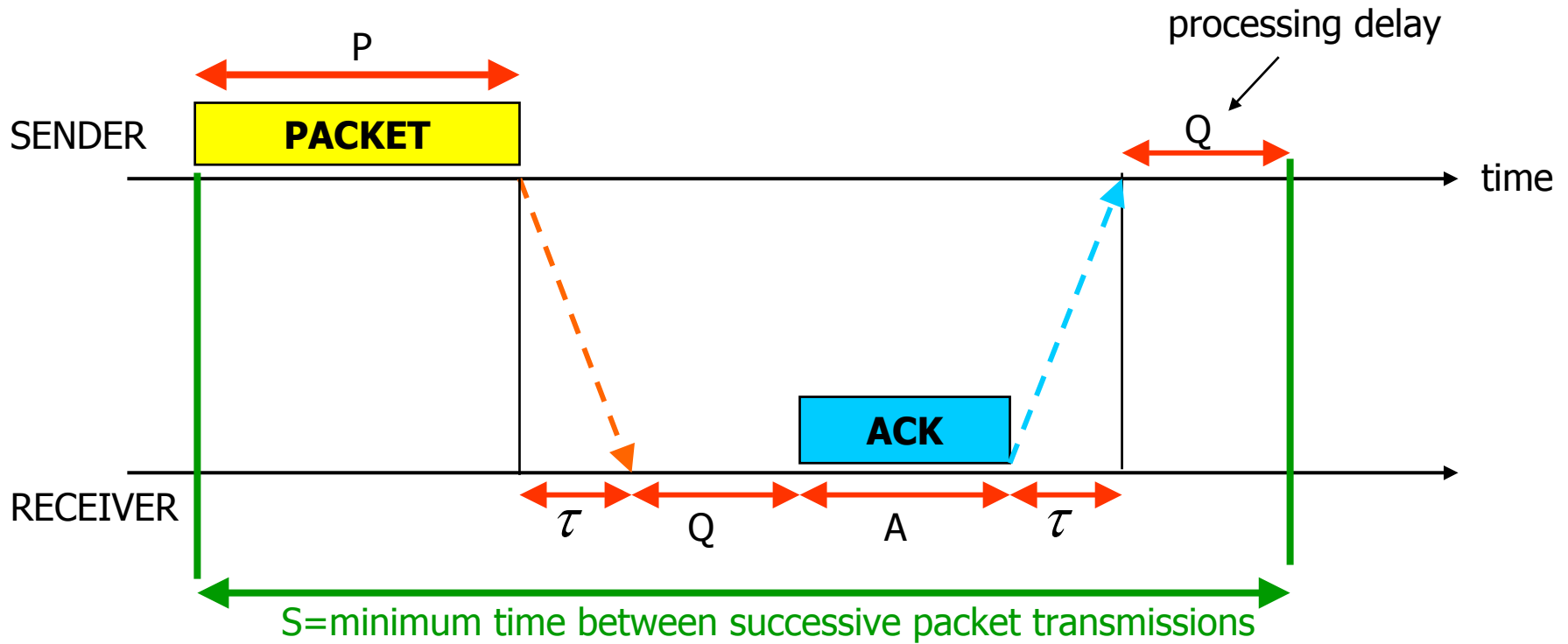
# Extra Slides

- Slides on protocol efficiency
  - ▣ For your enjoyment and edification..
  - ▣ This information will not be on the exams

# Efficiency of SWP

# Efficiency of SWP

- Consider the case in which no errors occur.

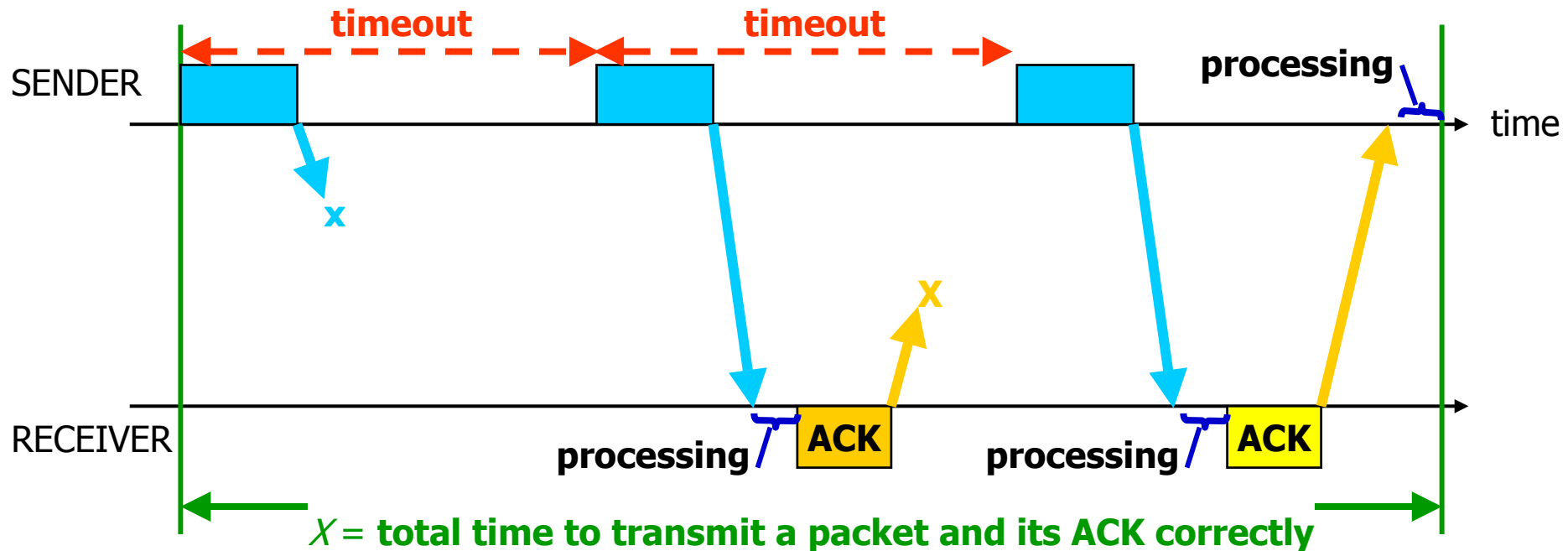


# Efficiency of SWP

From the diagram we have:

$$S = P + A + 2(Q + \tau)$$

However, packets and ACKs may be lost or corrupted in the channel and it takes much longer to send a packet successfully.

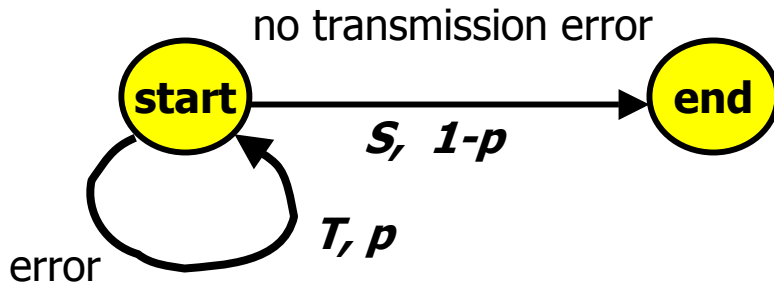


# Efficiency of SWP

- Let  $(1-p)$  be the probability that a packet or its ACK is sent sent correctly over the link.
- This implicitly assumes that each attempt to succeed sending a packet is independent of others.
- We have again the geometric random variable!
- The number of transmissions needed for success is of course  $1/(1-p)$ .
- All the failed transmissions incur *one timeout* of time, and all of these occur with probability  $p$ .
- The last transmission incurs  $S$  seconds and occurs with probability  $1-p$ .

# Efficiency of SWP

- We can apply the same simple method used to compute average delays in MAC protocols.



$T$  = timeout  
 $S$  =  $P+A+2Q+2\tau$

$$E\{X\} = (1-p)S + p(T + E\{X\})$$

$$E\{X\} = S + \frac{pT}{1-p}$$

$$\text{Efficiency} = \frac{\text{packet time}}{E\{X\}} = \frac{P}{E\{X\}}$$

$$\eta = \frac{(1-p)P}{(1-p)S + pT}$$

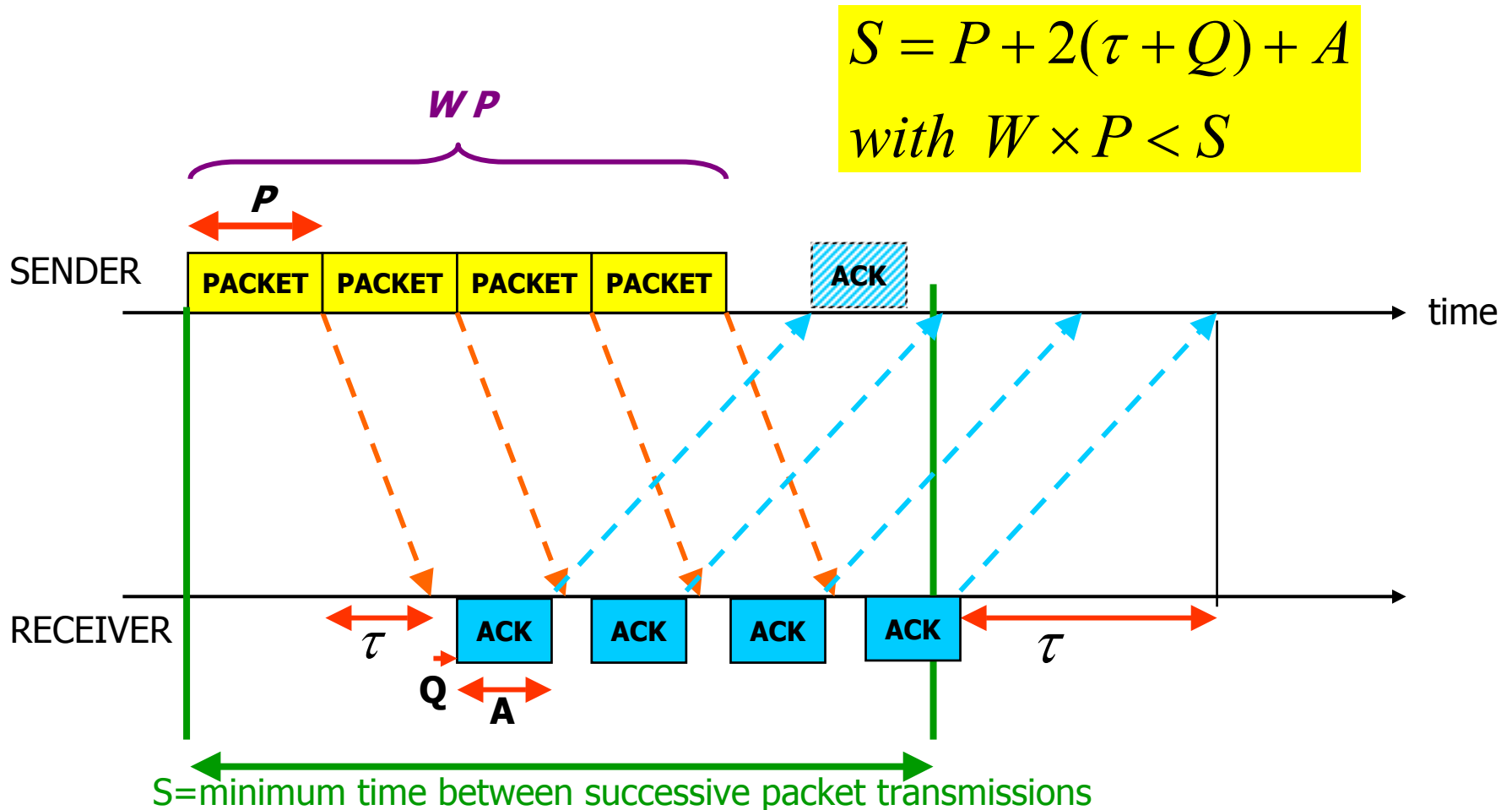
**Too much overhead over lossy links or links with large b-d products!**

$$\eta \approx \frac{(1-p)P}{(1-p)(P + A + 2\tau) + pT}; Q = 0$$

# Efficiency of SRP

# Efficiency of SRP

- Consider the case in which no errors occur.

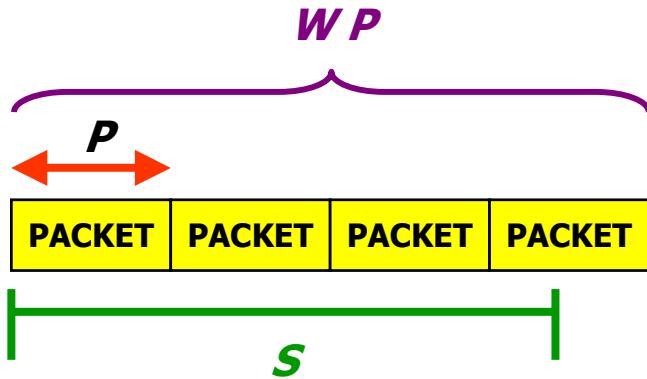




# Efficiency of SRP

- Efficiency of SRP:

$$\eta_{SRP} = \min \left\{ \frac{WP}{S}, 1 \right\}$$



With no errors,  
sender never stops with  $S < WP$

Assume **infinite**  $W$  and  $T > S$ , so that sender never retransmits if there are no errors.

$p = P\{\text{transmission error in packet or its ACK}\}$

$1-p = P\{\text{transmission successful packet and ACK}\}$

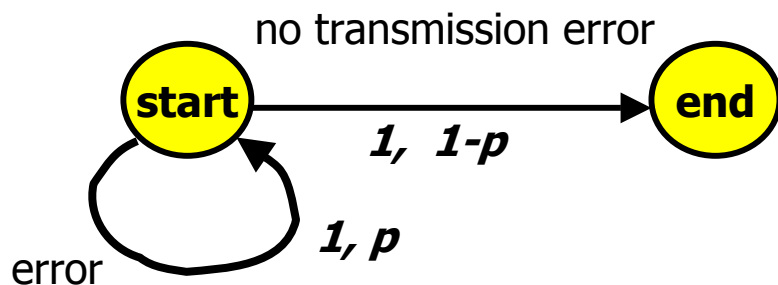
$Y =$  number of times packet is transmitted

*The average number of transmissions before success is again:*

$$E(Y) = \frac{1}{1-p}$$

# Efficiency of SRP

- We can also use the graphical method, of course...
- We focus on the number of transmissions, because all transmissions last the same.



$$E\{Y\} = (1-p) + p(1 + E\{Y\})$$

$$E\{Y\} = \frac{1}{1-p}, \text{ as expected!}$$

Let  $E(X)$  be the average time taken to transmit a packet successfully, then  $E(X) = P E(Y)$

$$\text{Efficiency} = \frac{\text{packet time}}{E\{X\}} = \frac{P}{E\{X\}}$$

$$\eta_{SRP} = \frac{P}{P/(1-p)} = 1-p \text{ for } W = \infty$$

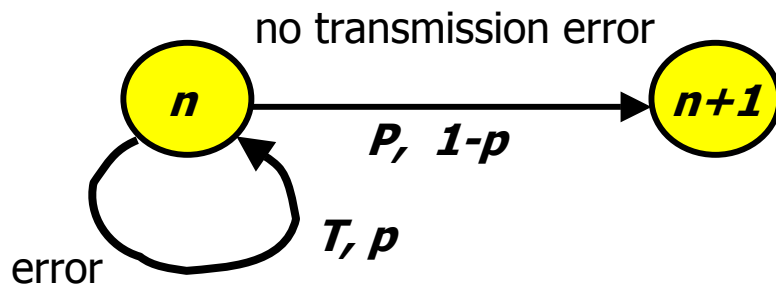
Very high efficiency as long as window is longer than  $S$ !

**Intuitive result! If sender never stops, the efficiency is the % of packets sent w/o errors**

# Efficiency of GBN

# Efficiency of GBN

- A packet that is not ACKed wastes a timeout ( $T$ )
- To simplify our analysis, assume  $T = S = W \times P$
- We compute the average time needed for the sender to get the ACK to a given packet  $n$  and advance window to  $n+1$



$$E\{X\} = (1-p)P + p(T + E\{X\})$$

$$E\{X\} = P + \frac{pT}{1-p} = P \left( 1 + \frac{pW}{1-p} \right)$$

$$\text{Efficiency} = \frac{\text{packet time}}{E\{X\}} = \frac{P}{E\{X\}}$$

$$\eta_{GBN} = \frac{1}{\left( 1 + \frac{pW}{1-p} \right)}$$