CN (IT-3001)

Transport Layer: Error and flow control Protocols

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University

Disclaimer: *The contents in this slide have been referred from many sources which I do not claim as my own. Some of the content has been modified for easier understanding of the students without any malafide intention. This slide is only for educational purpose strictly, and not for the commercial purpose.*

Transport Layer Protocols

- These protocols are implemented in the transport layer to provide basically two main functionality:
  1. Flow control and
  2. Error Control

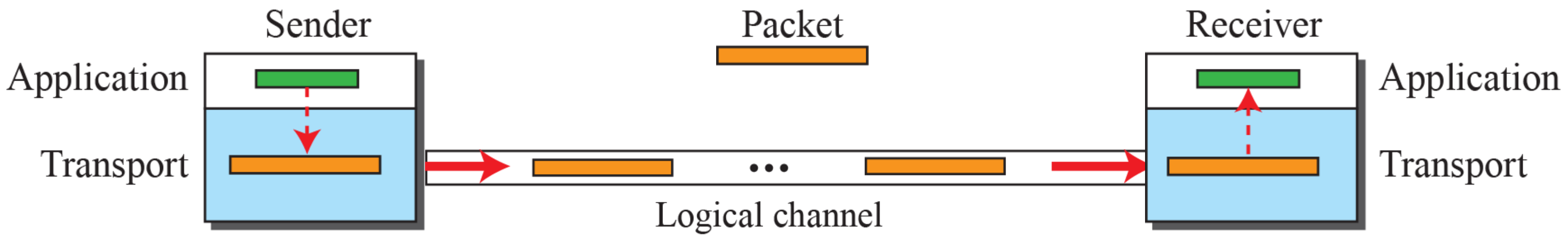
We discuss following protocols in the subsequent slides.

1. Simple Protocol
2. Stop-and-Wait Protocol
3. Go-Back-*N* Protocol
4. Selective-Repeat Protocol

Simple Protocol→ Connection Less Protocol

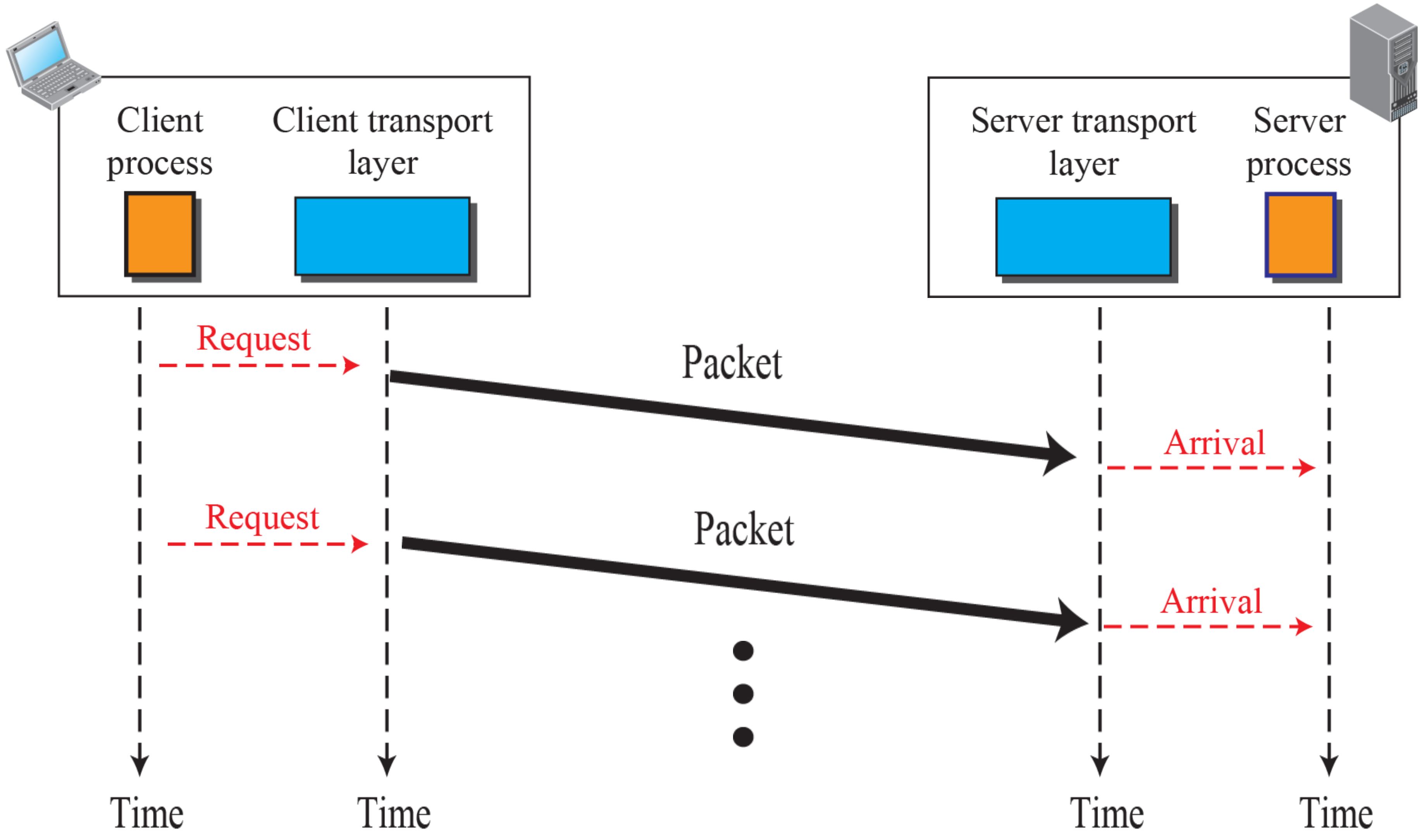
The channel is assumed to be ideal i.e., no error. The Sender and receiver are assumed to work at the same speed thus, no overwhelming of the receiver.

Thus, this protocol is designed for neither flow nor error control.



- **The Transport layer at the sender side**
    - gets a message from its application layer.
    - Then it makes a packet out of it, and sends the packet.
- **The Transport layer at the receiver side**
    - Receives a packet from its network layer
    - Extracts the message from the packet and delivers the message to its application layer.

Example 3.1: The sender sends packets one after another without even thinking about the receiver.



- Practically this can never be true, and thus the simple protocol is not used at all in practice.

- **Now, let us assume that the sender and the receiver do not operate at the same speed.**

  So, we need to design a protocol for flow control

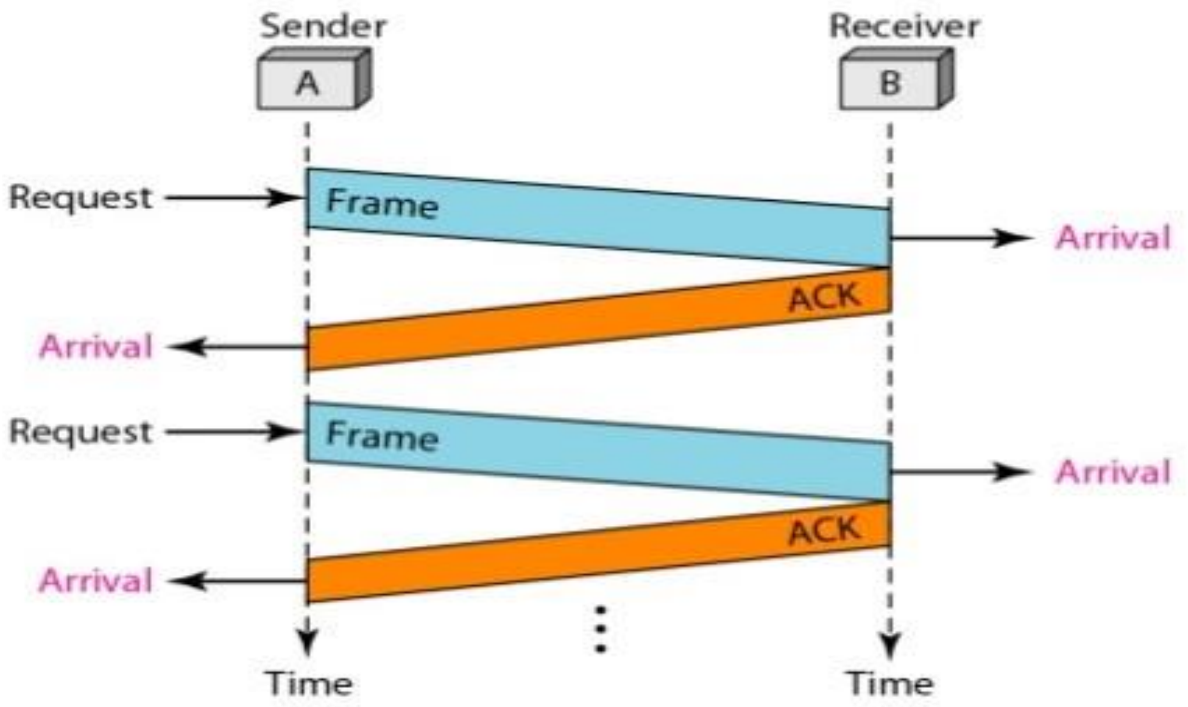
Note: *Here also, we assume that the channel is ideal i.e., there is no error.*

Stop-and-Wait Protocol

- This is connection oriented protocol.
- Here, we implement **only** flow control.
- The flow control is implemented using following strategy:
    - The sender must wait until it receives the **ACK** packet before sending the next packet.

Note: *A small processing delay may be introduced between reception of the last byte of a Data PDU and generation of the corresponding ACK.*

Example 3.2: The sender sends one frame and waits for feedback from the receiver. **If and only if** the ACK arrives, the sender sends the next frame.



Note: *The sending two frames in the protocol involves the sender in four events and the receiver in two events.*

- Practically, noiseless channel does not exist.
- So, we need to design a protocol for noisy channel.
- The Stop-and-Wait protocol can also be used for flow control in noisy channel with some modification for error control as well. It can use the checksum for detecting the corrupted packets.
- So, for noisy channel, we have following three protocols implementing flow as well as error control.
    1. *Stop-and-Wait Protocol*
    2. *Go-Back-N Protocol*
    3. *Selective-Repeat Protocol*

For non-ideal channel, in general, the following two cases may occur….

1. If Packet is
   - Corrupted: To detect corrupted packets, add checksum to data packet.
   - Lost: resend the packet
   - Out of order: Give numbering to each packet, called sequence number
2. If ACK is
   - Corrupted: add redundant bits i.e., checksum.
   - Lost: give numbering to each ACK, called ACK number
   - Out of order: give numbering

Stop-and-Wait Protocol

- It is connection oriented protocol that implements both error and flow control.
- Error control is implemented by adding checksum to each packet.
- When the packet arrives at the receiver site, it is checked; and if it is corrupted, it is silently discarded.
- Lost packets are more complex to handle than corrupted one.
- Unlike previous protocols, each packet is numbered for identification.
- The corrupted and lost packet are resent as follows.
  - The sender keeps a copy of each sent packet.
  - At the same time, it starts a timer.
  - If the timer expires and there is no ACK for the sent packet, the packet is resent, the copy is held, and the timer is restarted.

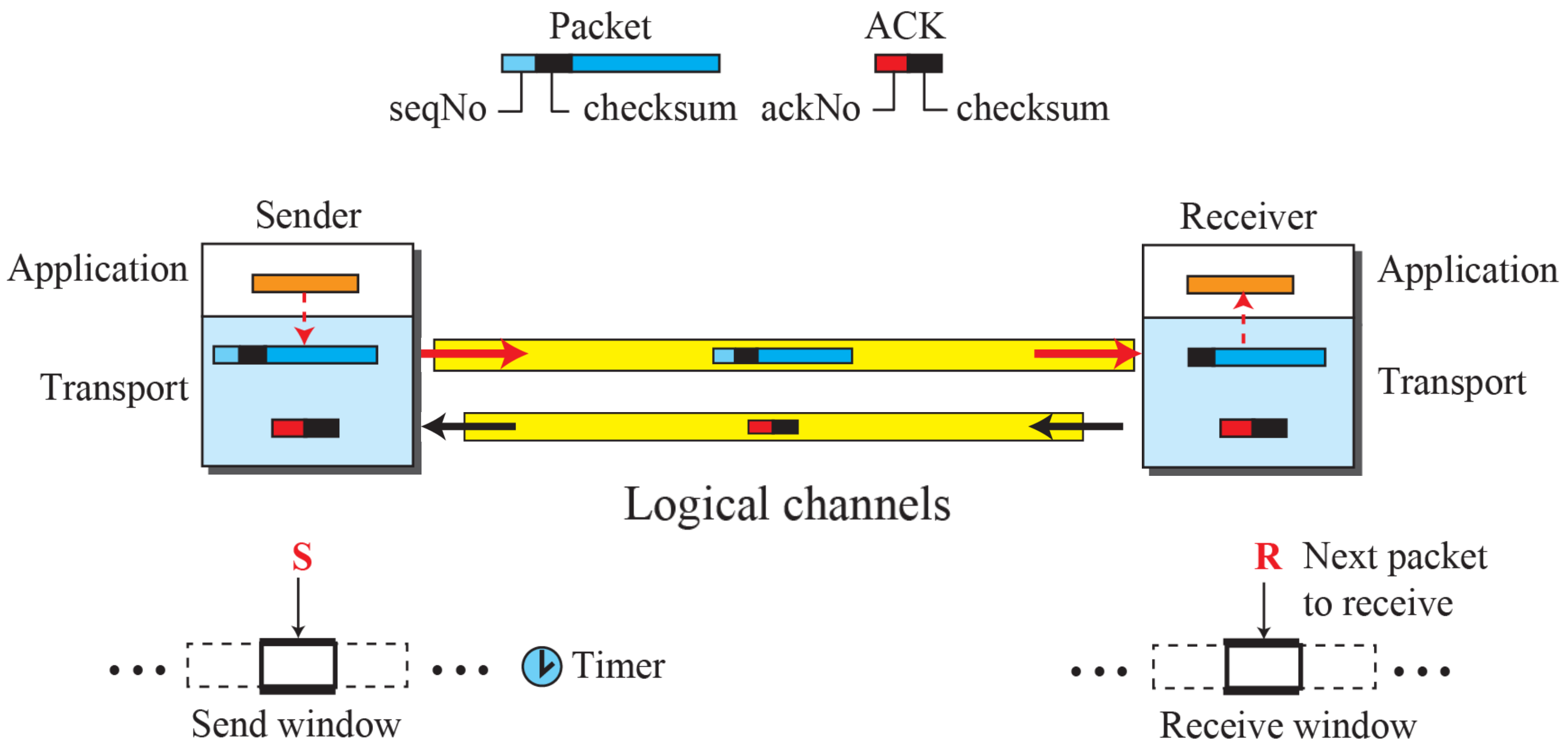
Stop-and-Wait Protocol

- **Sequence number (seqNo):** It is the packet being sent. Only 1-bit sequence number will suffice, i.e., either 1 or 0 can be the sequence number.
- This is bcz, only three things can happen when sender sends packet x.
    1. If packet arrives safe and sound at the Rx then receiver sends ACK x+1.
    2. If packet arrives safe and sound at the Rx, but ACK is lost or corrupted then sender resends packet x after timeout. Rx can recognize that this is a duplicate frame because it was expecting packet x+1 instead of x.
    3. The packet never reaches at the receiver site, sender resends the packet x.

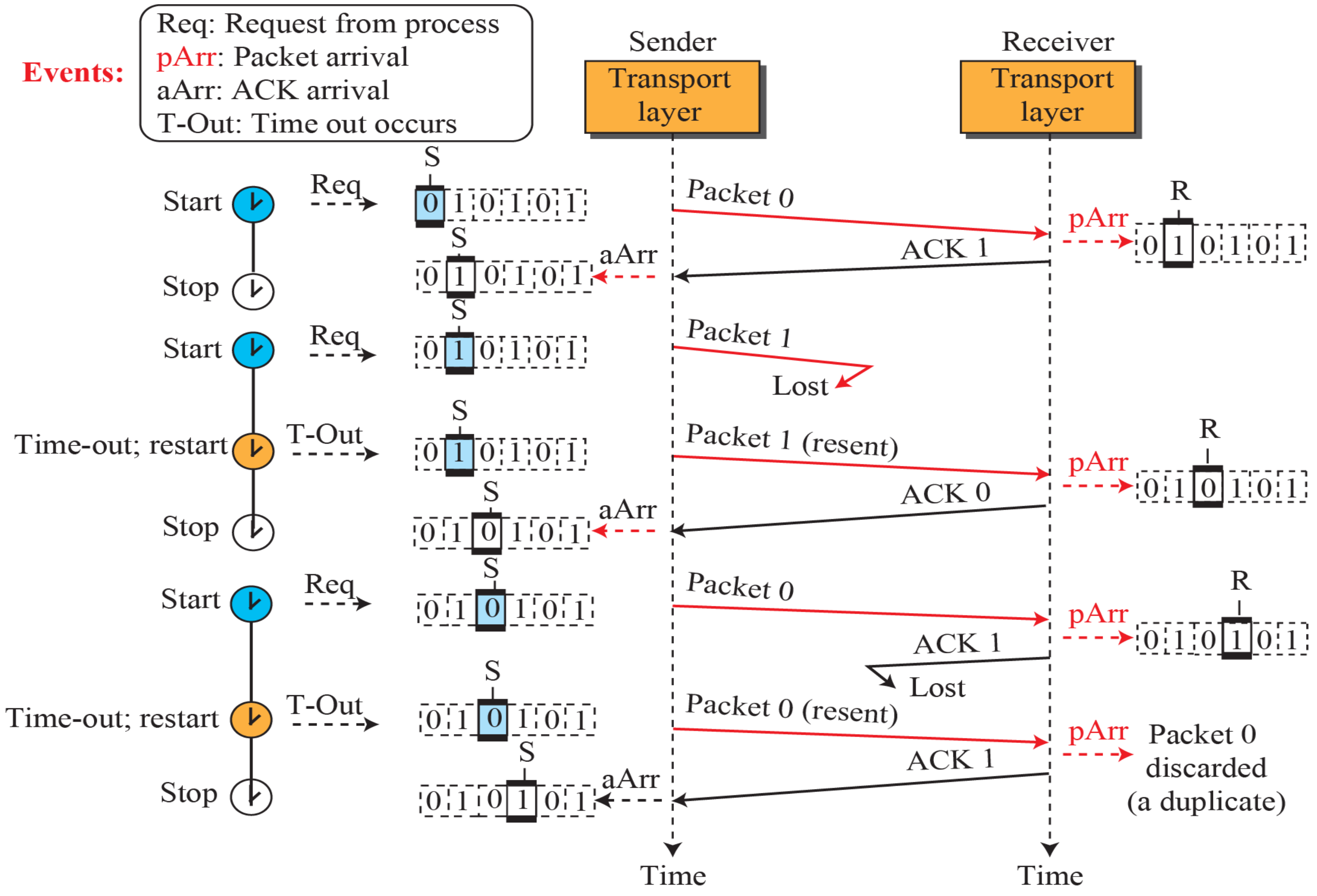
- **Acknowledgement number (ackNo):** It is the sequence number of the next expected packet.

For Stop-and-Wait Protocol, Illustration of

- Packets → seqNo and checksum
- ACK → ackNo and checksum
- Send and receive window size



Example 3.3: Packet 0 is sent and acknowledged. Packet 1 is lost and resent after the time-out. The resent packet 1 is acknowledged and the timer stops. Packet 0 is sent and acknowledged, but the acknowledgment is lost. The sender has no idea if the packet 0 or the acknowledgment 1 is lost, so after the time-out, it resends packet 0, which is acknowledged.



Stop-and-Wait ARQ: Key points

- **Advantage:**

- Easy to implement.
- Requires minimum buffer size comparatively as the send and receive window size is same and equal to 1.

- **Disadvantage:**

- It makes highly inefficient use of communication links.
- Particularly, when the channel is thick (more bandwidth) and long (needs long time for ACK to reach to sender).

Efficiency of Stop-and-Wait ARQ

- <u>**Example 3.3:**</u>
    - Assume that, in a Stop-and-Wait system, the bandwidth of the line is 1Mbps, and 1 bit takes 20ms to make a round trip. What is the bandwidth delay product? If the system data packets are 1000 bits in length, what is the utilization percentage of the link?

Sol: *The bandwidth delay product is*

$$= \ (\mathbf{1} \times \mathbf{10^6}) \times (\mathbf{20} \times \mathbf{10^{-3}}) = \mathbf{20{,}000} \boldsymbol{bits}$$

The system can send 20,000 bits but it is actually sending only 1000 bits (packet size). Thus , the link utilization factor is only 1000/20000, or 5%.

- **<u>Example 3.4:</u>**
  - What is the utilization percentage of the link with bandwidth 1Mbps, and 1 bit takes 20ms to make a round trip, if we have a protocol that can send up to 15 packets before stopping and worrying about the acknowledgments?
- **Sol:** *The bandwidth delay product is same i.e., 20,000 bits.*

But now a packet consists of 1000 bits and at a time we can send up to 15 packets. So, total number of bits during a round trip is….

15,000/20,000, or 75%

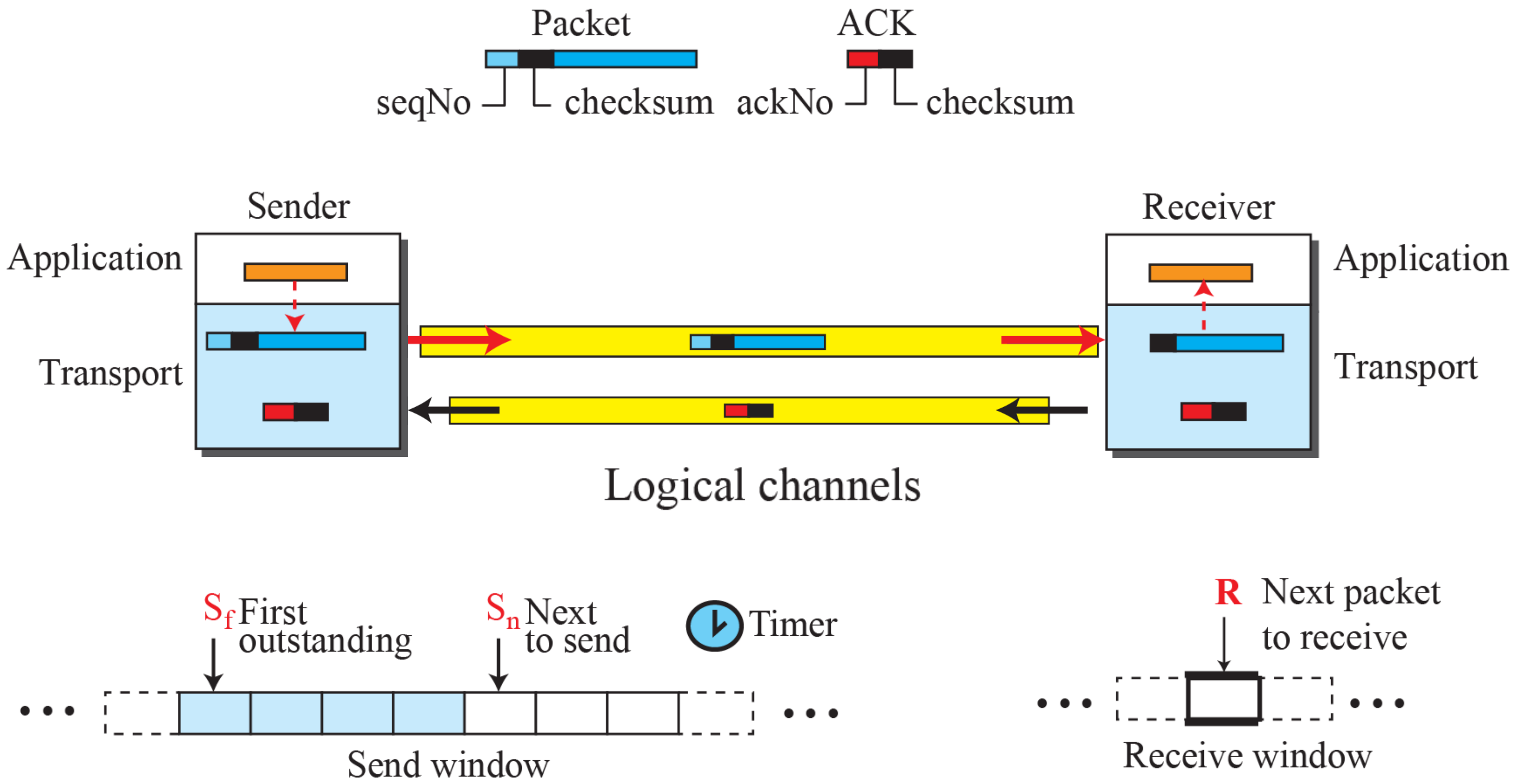
<u>Conclusion:</u> The efficiency of the system can be increased if instead of sending only one packet at a time and waiting for acknowledgement, we can send more than one packet without waiting for the acknowledgement.

Go-Back-*N* (GBN)

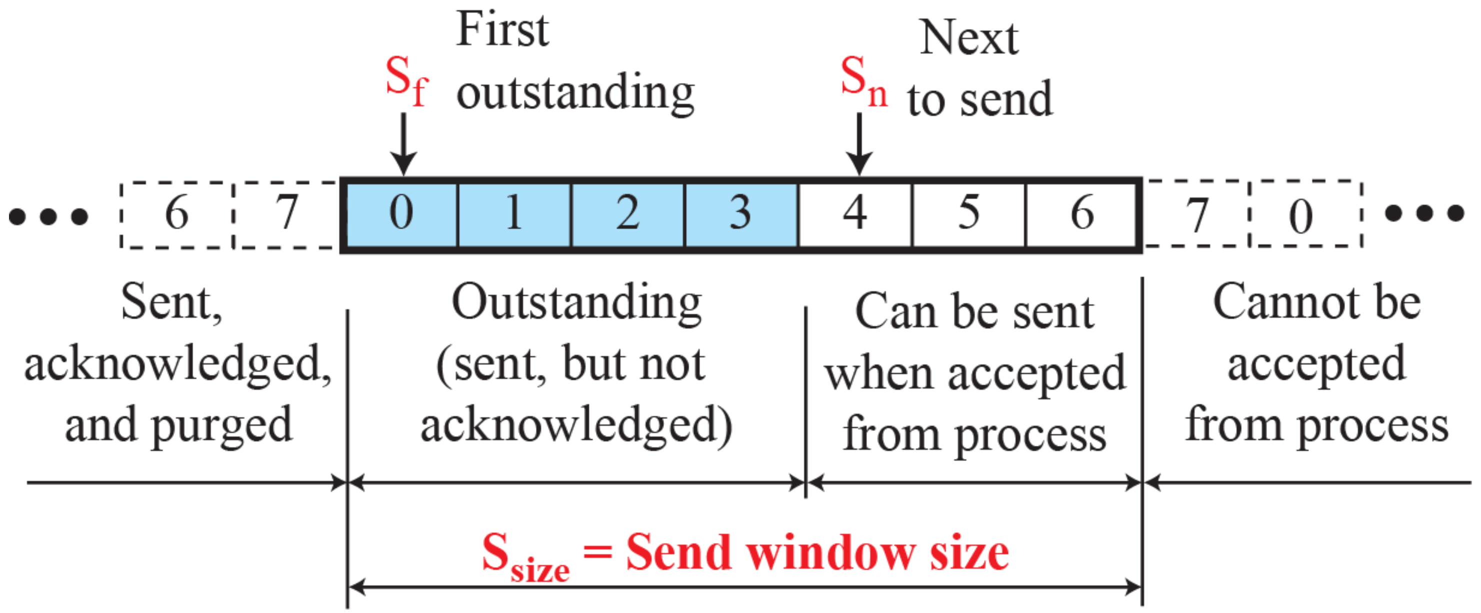
- **Key idea:** To improve link efficiency (filling the pipe) of Stop-and-Wait, multiple packets must be in transition while waiting for acknowledgement.
- In Go-Back-N protocol, several packets are sent before receiving acknowledgement.
- **Sequence number:** Because of multiple packets, one bit sequence number will not suffice. Generally if header allows *m-bits* for the sequence number, the sequence number ranges from 0 to $2^m - 1$, i.e. sequence numbers are modulo 2^m. Sequence number can be repeated.
- **Acknowledgement number:** ackNo in this protocol is **cumulative** and defines the sequence number of the next packet expected.

For Go-Back-*N* Protocol, Illustration of

- Packets → seqNo and checksum
- ACK → ackNo and checksum
- Send and receive window size



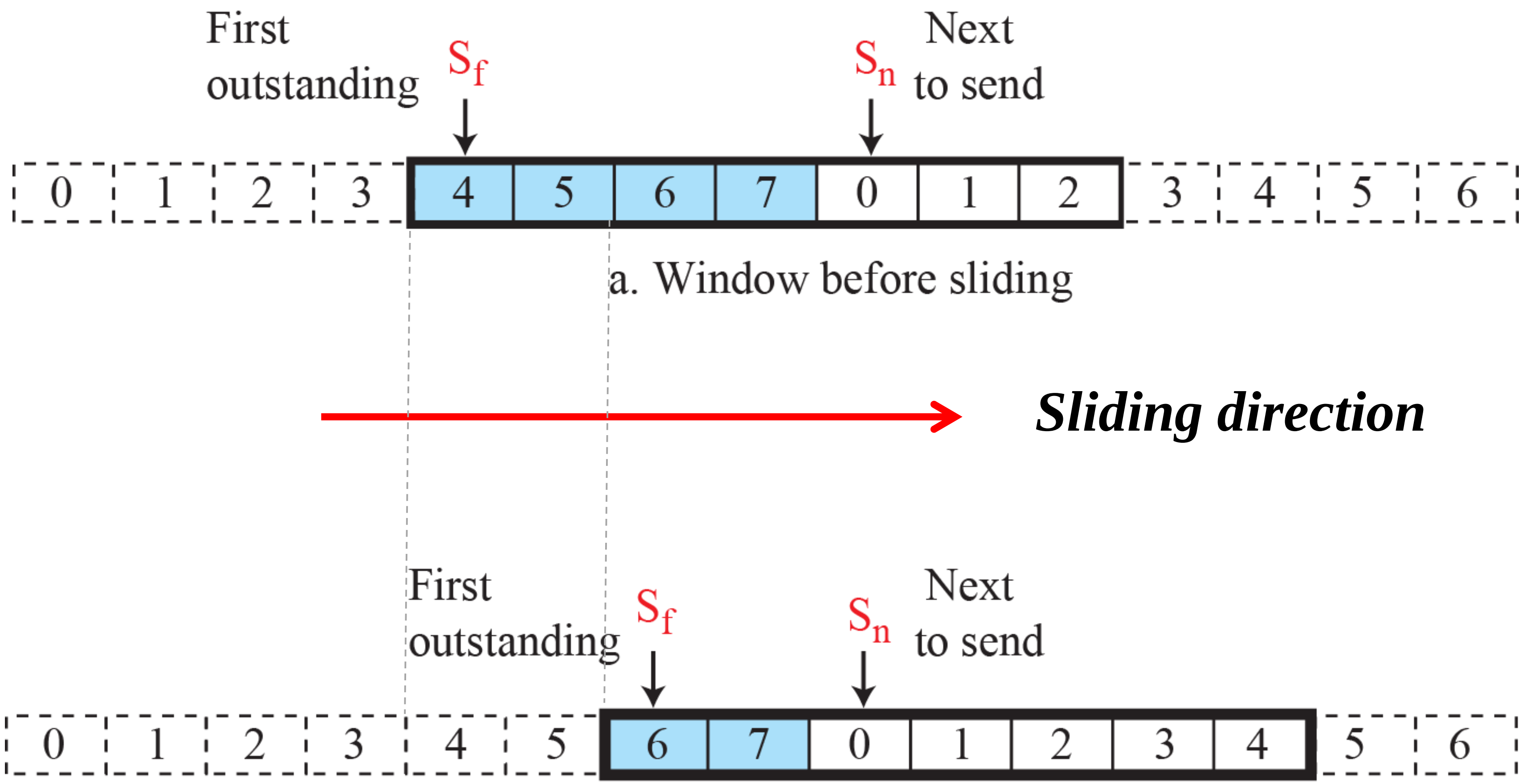
- **Sender Sliding Window:** It is an abstract concept defining an imaginary box of size $2^m - 1$ with three variables $S_f, S_n, and\ S_{size}$.



First outstanding
S_f
Next to send
S_n
6
7
0
1
2
3
4
5
6
7
0
Sent, acknowledged, and purged
Outstanding (sent, but not acknowledged)
Can be sent when accepted from process
Cannot be accepted from process
S_{size} = Send window size

Note: The value of m is 4 in aforementioned example.

Sliding of sending window



a. Window before sliding

b. Window after sliding (an ACK with ackNo = 6 has arrived)

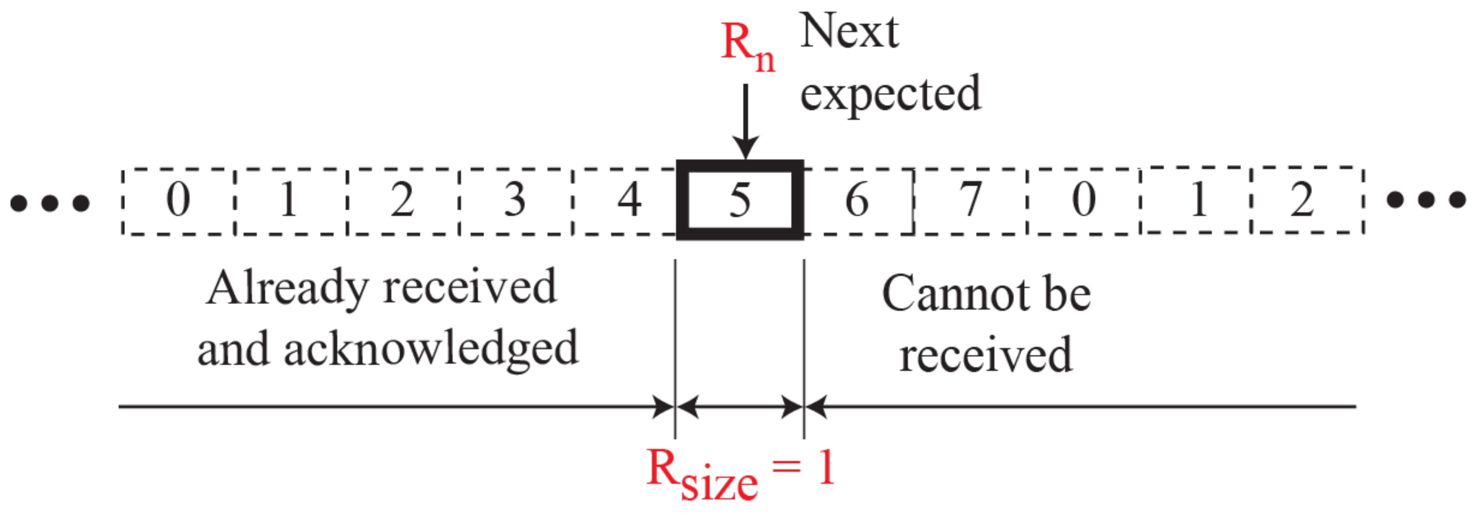
Note:- Send window can slide more than one slot

Please try out…….

- Repeat the last example, i.e., draw the receiver window by explicitly mentioning sequence number for ‘m=3’. Also, repeat for ‘m=5’.

- **Receiver Sliding Window:** The receive window is an abstract concept defining an imaginary box of **size 1** with one single variable R_n *(receive window, next packet expected).* The window slides when a correct packet has arrived; sliding occurs one slot at a time.

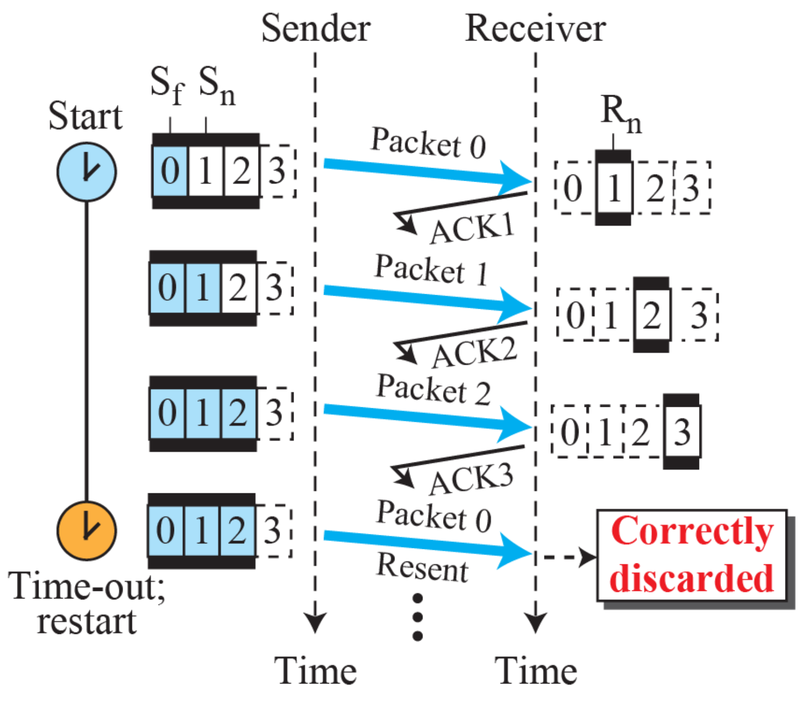
- Only a packet with a sequence number matching the value of R_n is accepted and acknowledged.



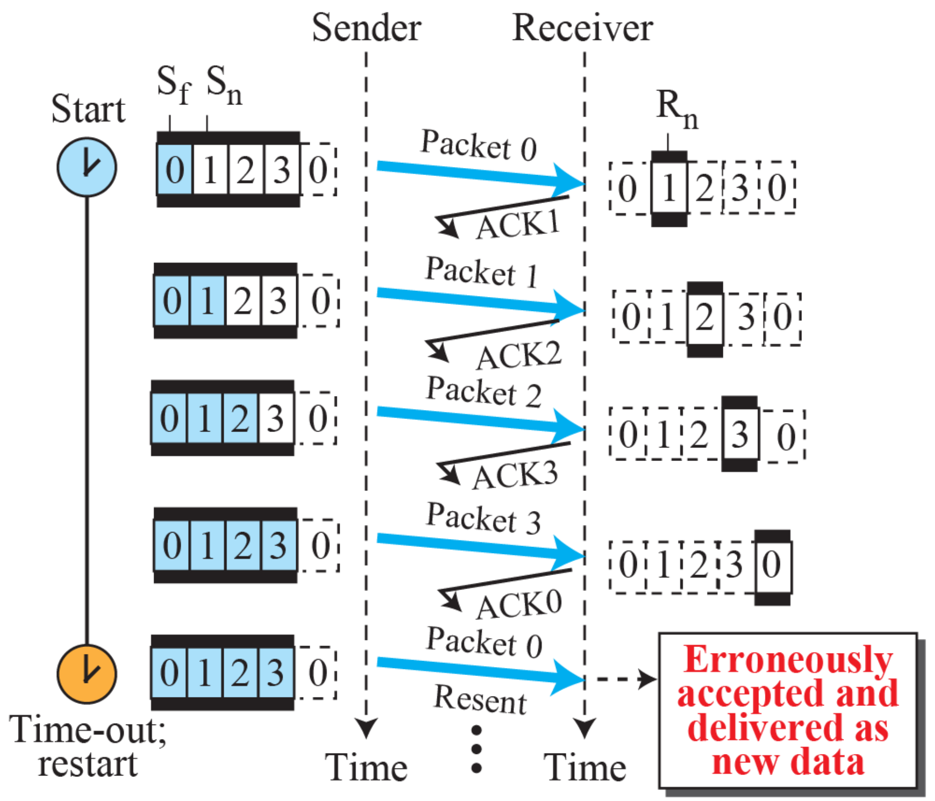
Go-Back-*N*: Key points

- **Timers:** instead of having timer for each frame, we always keep timer **for the first outstanding frame**. This is because timer for the first outstanding frame always **expires first**. We send all outstanding frames when this timer expires.
- **Acknowledgement:** The receiver sends a positive acknowledgement if and only if a frame has arrived safe, sound, and **in order**. If a frame is damaged or is received out of order, the receiver is silent and will discard all subsequent frames until it receives the one it is expecting.
- **Resending frame:** Whenever the timer expires, the sender resends **all outstanding frame**. This is why this protocol is known as Go-Back-**N** ARQ

Why to keep sender window size $2^m - 1$?

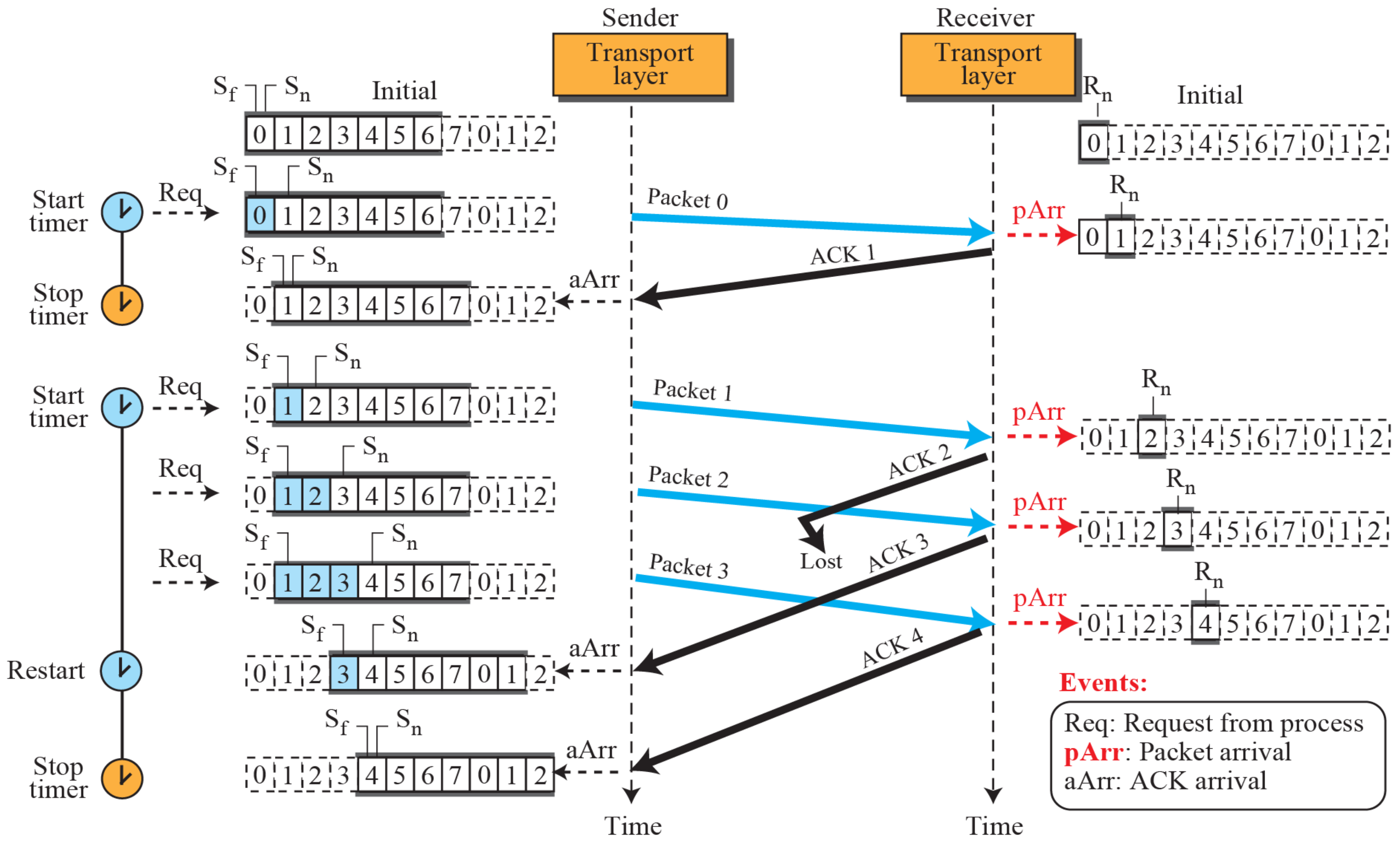


a. Send window of size < 2^m



b. Send window of size = 2^m

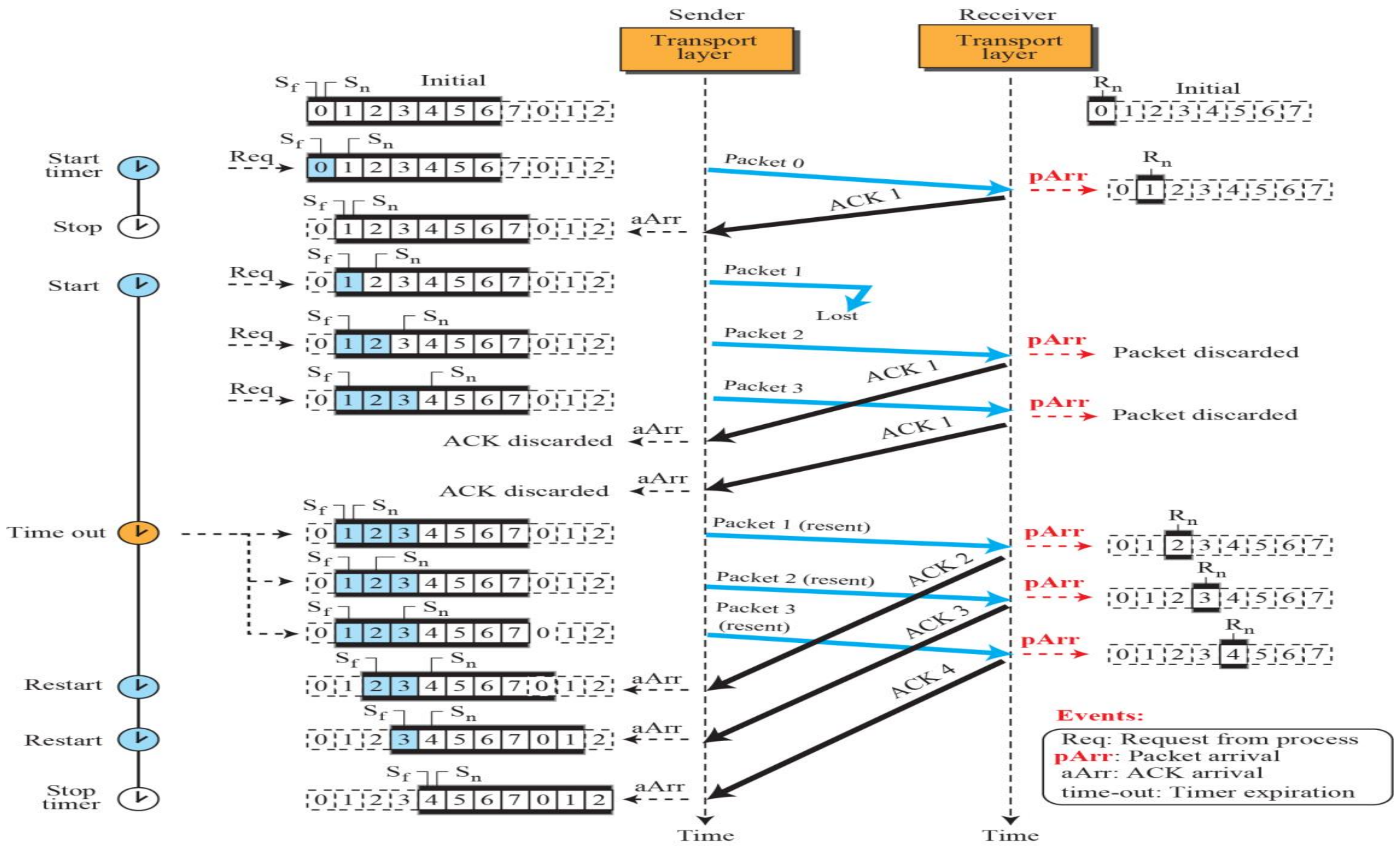
Example 3.5: Figure below shows an e.g. of Go-Back-N ARQ where the forward channel is reliable, but the revers is not. No packets are lost, but some ACKs are delayed and one is lost. This example also shows how cumulative acknowledgements can help if acknowledgements are delayed or lost.



Go-Back-N Vs. Stop-and-Wait

Stop-and-Wait	Go-Back-N
Sender window size is 1	Sender window size is $2^m - 1$
Receiver window size is 1	Receiver window size is 1
1 bit sequence number i.e., either 0 or 1	m bit sequence number i.e., from 0 to $2^m - 1$

Example 3.6: For the example shown below, write all the events with description in your own words.



- **Hint for Example 3.6:**
  - Packets 0, 1, 2 and 3 are sent.
  - However, packet 1 is lost. The receiver receives packet 2 and 3, but they are discarded because they are received out of order.
  - When the receiver receives packet 2 and 3, it sends ACK 1 to show that it expects to receiver packet 1.
  - However, these ACKs are not useful for the sender because the ackNo is equal to S_f, and not greater than S_f. So the sender discards them.
  - When the timeout occurs, the sender resends packets 1, 2 and 3, which are acknowledged.

Is this an efficient method to resend all N-packets even though there is error in any single packet out of N-packets?

Is this an efficient method to resend all N-packets even though there is error in any single packet out of N-packets?

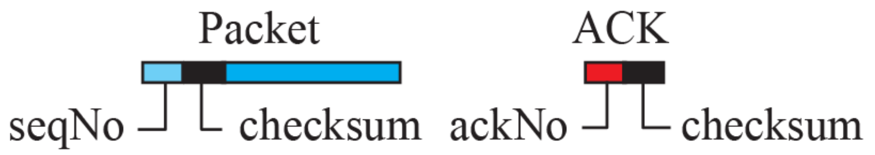
Ans: Obviously no…*otherwise we may waste some link bandwidth.*

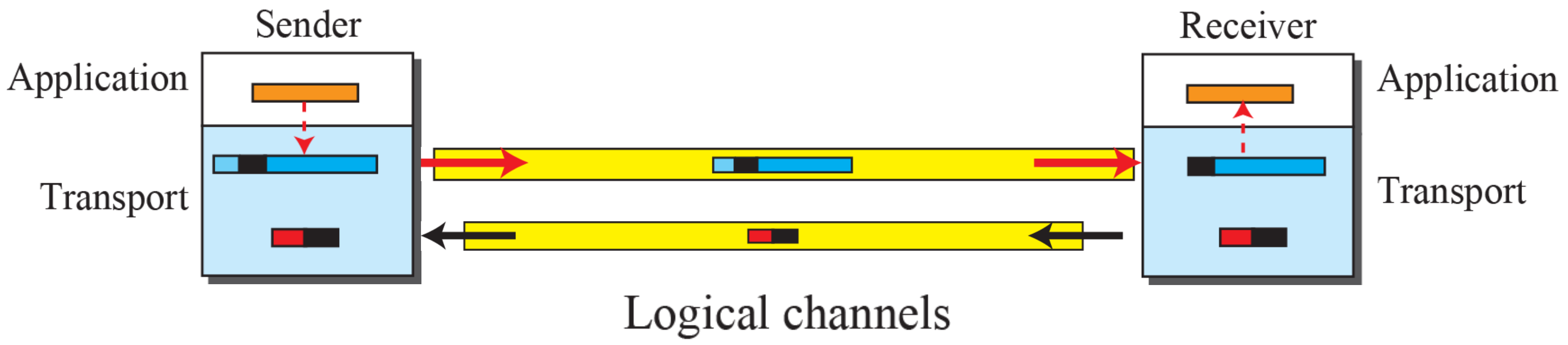
Selective-Repeat Protocol

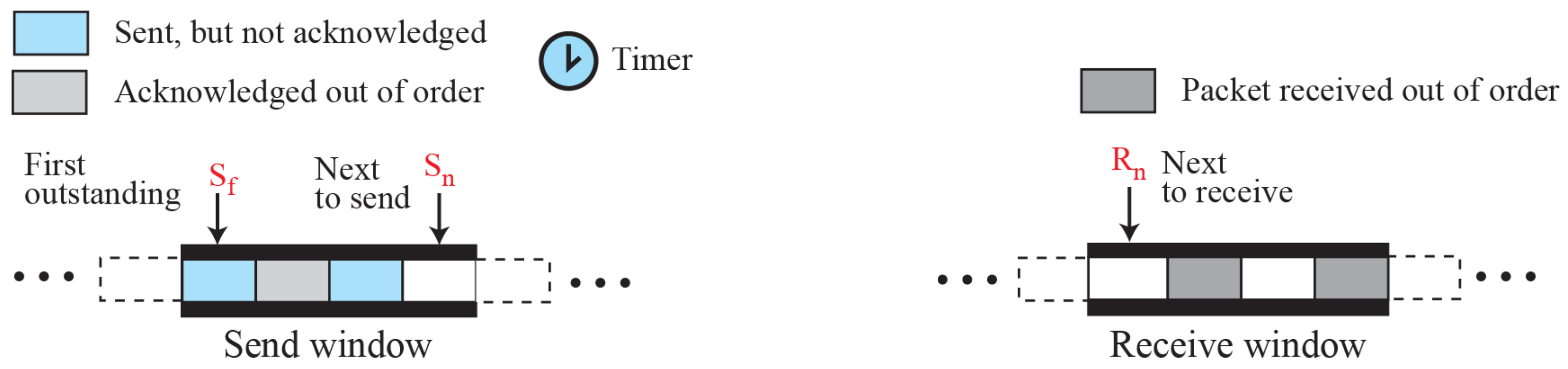
- In this protocol, receive window and send window are of same size.
- The maximum window size is much smaller than one used in Go-Back-*N*. Here it is 2^{m-1}.
- E.g., for m=4 bit sequence number, sequence number go from 0 to 15 but the window size is only 8.
- The smaller window size means less efficiency in filling the pipe, **but** the fact that there are fewer duplicate packets can compensate for this.

For Selective Repeat Protocol, Illustration of

- Packets → seqNo and checksum
- ACK → ackNo and checksum
- Send and receive window size

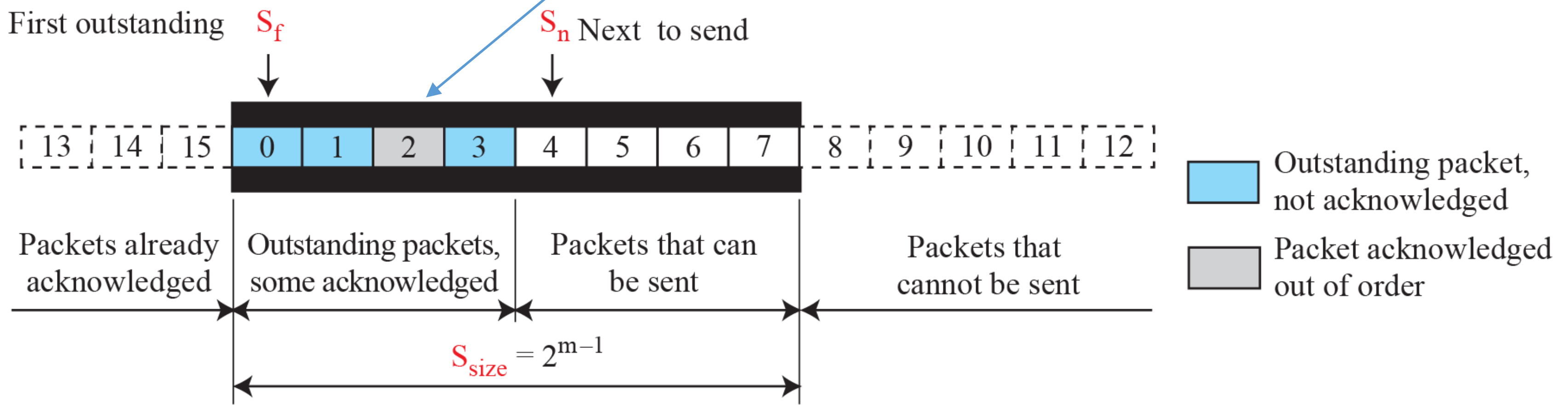






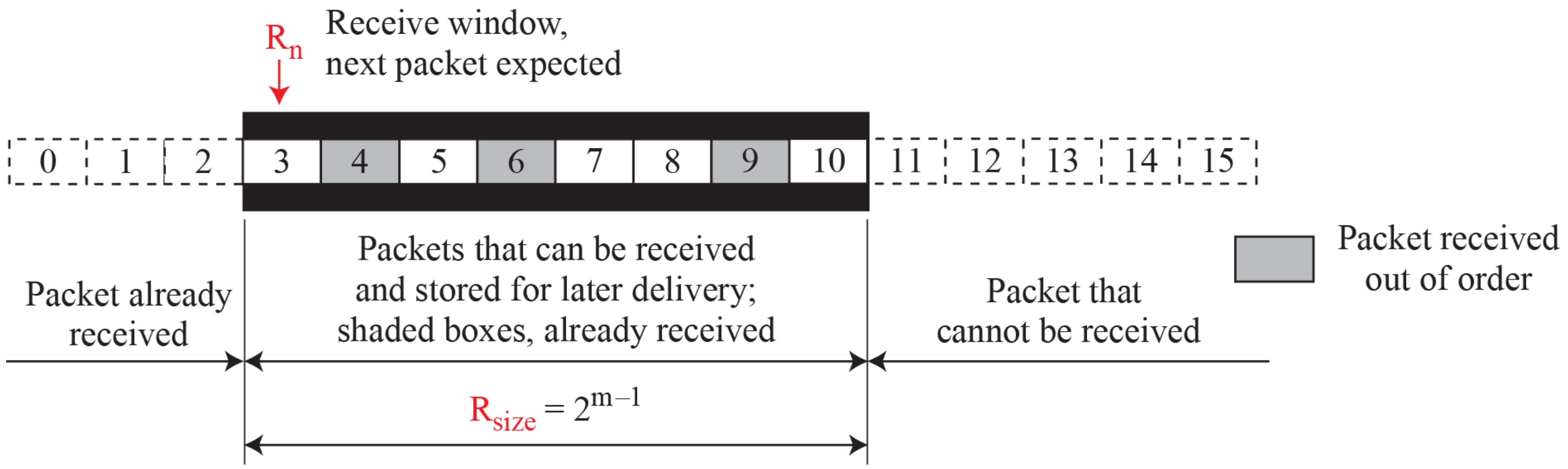
- **Sender Sliding Window:** It is an abstract concept defining an imaginary box of size 2^{m-1} with three variables $S_f, S_n, and\ S_{size}$.

This slots indicate that packet arrived out of order and waiting for their neighbours to arrive before delivery to the Application layer


First outstanding
S_f
S_n Next to send
13
14
15
0
1
2
3
4
5
6
7
8
9
10
11
12
Outstanding packet, not acknowledged
Packet acknowledged out of order
Packets already acknowledged
Outstanding packets, some acknowledged
Packets that can be sent
Packets that cannot be sent
$S_{size} = 2^{m-1}$

Note: The value of m is 4 in aforementioned example.

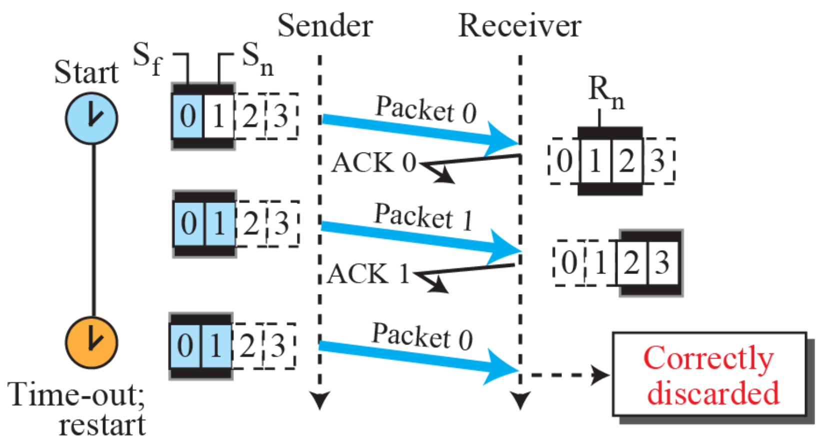
Illustration of Receiving Window for Selective-Repeat



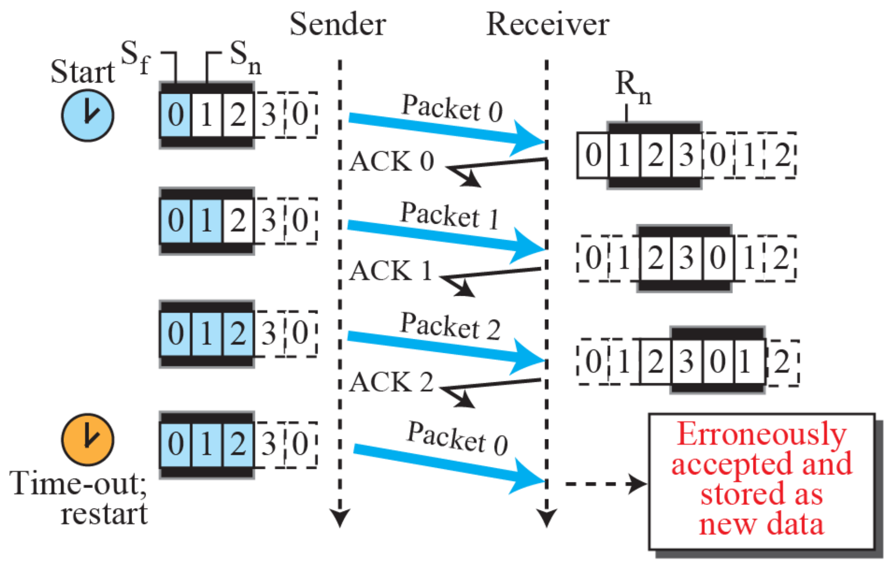
R_n
Receive window, next packet expected
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Packet already received
Packets that can be received and stored for later delivery; shaded boxes, already received
Packet that cannot be received
Packet received out of order
$R_{size} = 2^{m-1}$

Why window size is 2^{m-1} only?

- Let m=2, i.e., total four sequence numbers are 0, 1, 2, 3, Now, we choose window size 2 and 3 for analysis.

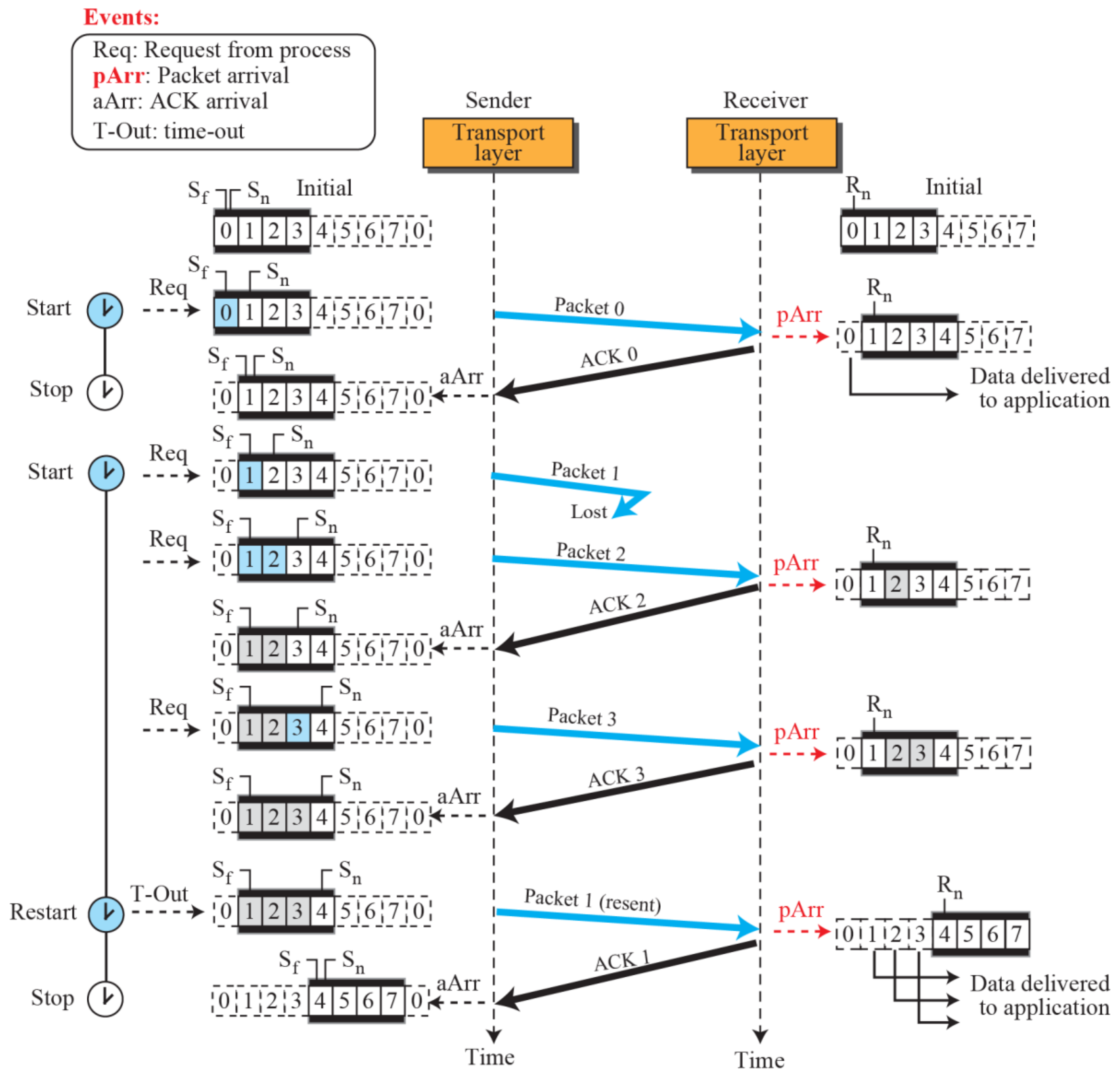


a. Send and receive windows of size = 2^{m-1}



b. Send and receive windows of size > 2^{m-1}

Example 3. 7: It is similar to Example 3.6. Write description of all events in your own words.



Only packet 1 resent

Example 3.8: Assume a sender sends 6 packets: packets 0, 1, 2, 3, 4, and 5. The sender receives an ACK with ackNo = 3. What is the interpretation if

1. The system uses Go-Back-*N* protocol
2. The system uses Selective-Repeat protocol?

Solution

- If the system uses Go-Back-*N* protocol, it means that packets 0, 1, and 2 have been received uncorrupted and the receiver is expecting packet 3.
- If the system uses Selective Repeat protocol, it means that packet 3 has been received uncorrupted; the ACK does not say anything about other packets.

Now, Think….

- *An acknowledgement number in the Go-Back-N protocol defines the next packet expected, but an acknowledgement number in the Selective-Repeat protocol defines the sequence number of the packet to be acknowledged. Can you explain the reason?*

Key Points

Stop-and-Wait	**Go-Back-*N***	**Selective-Repeat**
Sender window size is 1	Sender window size is $2^m - 1$	Sender window size is $\mathbf{2^{m-1}}$
Receiver window size is 1	Receiver window size is 1	Receiver window size is also $\mathbf{2^{m-1}}$
1 bit sequence number i.e., either 0 or 1	m bit sequence number i.e., from 0 to $2^m - 1$	m bit sequence number i.e., from 0 to $2^m - 1$

Please try out…….

- Calculate sequence number, sender window size, receiver window size, and draw the sender and receiver window explicitly for ‘m=3’. Also, repeat for ‘m=4’ and ‘m=5’.

Note: While explaining any of the three protocol discussed for noisy channel, please keep in mind the following questions……….

1) What is the range of sequence number and ACK number?
2) What is the significance of ACK number?
3) What is the size of sender window and receiver window?
4) What should be the position of S_f, S_n, R_n after every operation (sending frames and receiving ACK or NAK)?
5) Show that how does sender or receiver or both window slide explicitly.
6) Show sender and receiver window explicitly by drawing overlapping rectangle.
7) Show timer time out conditions as and when requires.

What is piggybacking?

Piggybacking

Disclaimer:- *The images are taken from the Google for explaining the concept to students. Its intention is not to hurt sentiments of any students or person portrayed in this.*

Piggybacking

- The three protocols we discussed in this section are all unidirectional: data packets flow in only one direction although control information such as ACK and NAK packets can travel in the other direction.
- In real life, data packets are normally flowing in both directions. This means that the control information also needs to flow in both directions.
- A technique called **piggybacking** is used to improve the efficiency of the bidirectional protocols.
- When a packet is carrying data from A to B, it can also carry control information about arrived (or lost) packets from B; when a packet is carrying data from B to A, it can also carry control information about the arrived (or lost) packet from A.

Disclaimer:- *The images are taken from the Google for explaining the concept to students. Its intention is not to hurt sentiments of any students or person portrayed in this.*