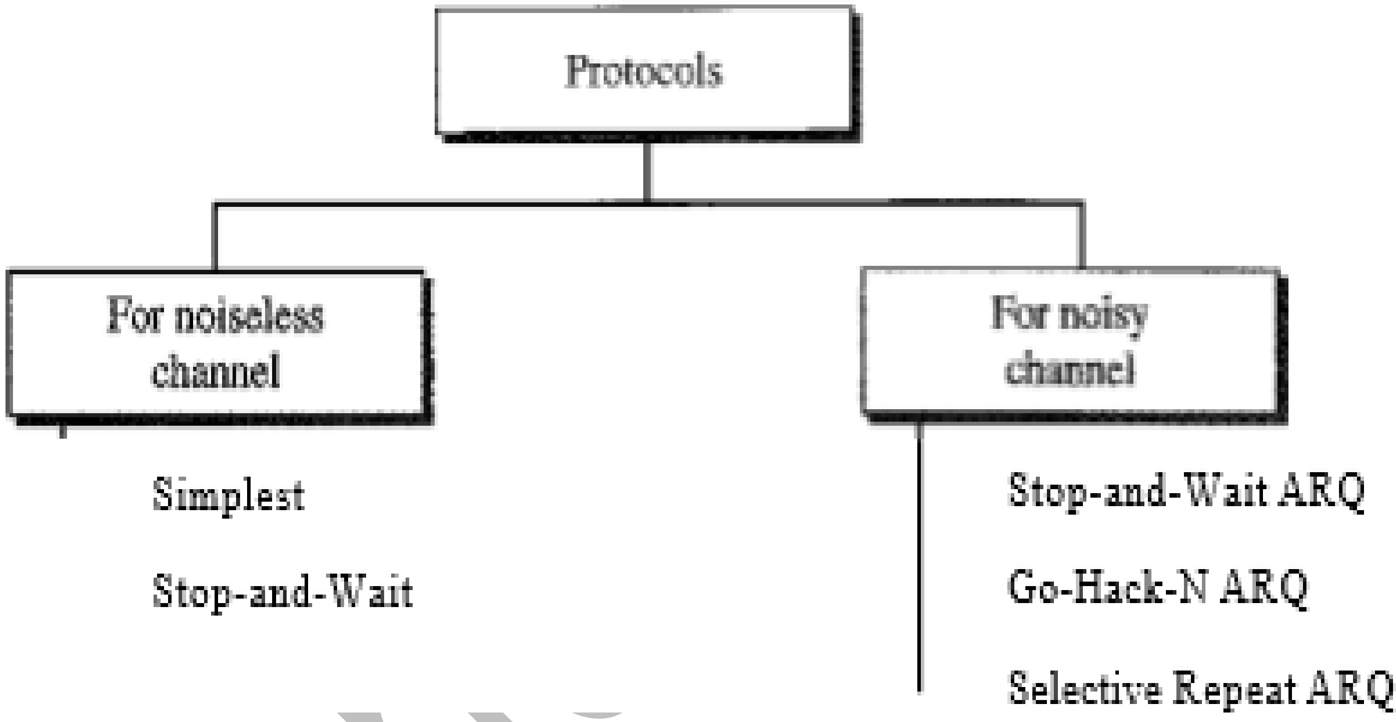
FLOW AND ERROR CONTROL

- Flow control coordinates the amount of data that can be sent before receiving an acknowledgment and is one of the most important duties of the data link layer.
- The flow of data must not be allowed to overwhelm the receiver. Any receiving device has a limited speed at which it can process incoming data and a limited amount of memory in which to store incoming data.
- The receiving device must be able to inform the sending device before those limits are reached and to request that the transmitting device send fewer frames or stop temporarily.
- Incoming data must be checked and processed before they can be used. The rate of such processing is often slower than the rate of transmission. For this reason, each receiving device has a block of memory, called a buffer, reserved for storing incoming data until they are processed.
- If the buffer begins to fill up, the receiver must be able to tell the sender to halt transmission until it is once again able to receive.

- **Error Control (lost, out of order, corrupt) (detection and retransmission)**
    - Error control is both error detection and error correction. It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender.
    - In the data link layer, the term error control refers primarily to methods of error detection and retransmission.
    - Error control in the data link layer is often implemented simply: Any time an error is detected in an exchange, specified frames are retransmitted. This process is called automatic repeat request **(ARQ)**

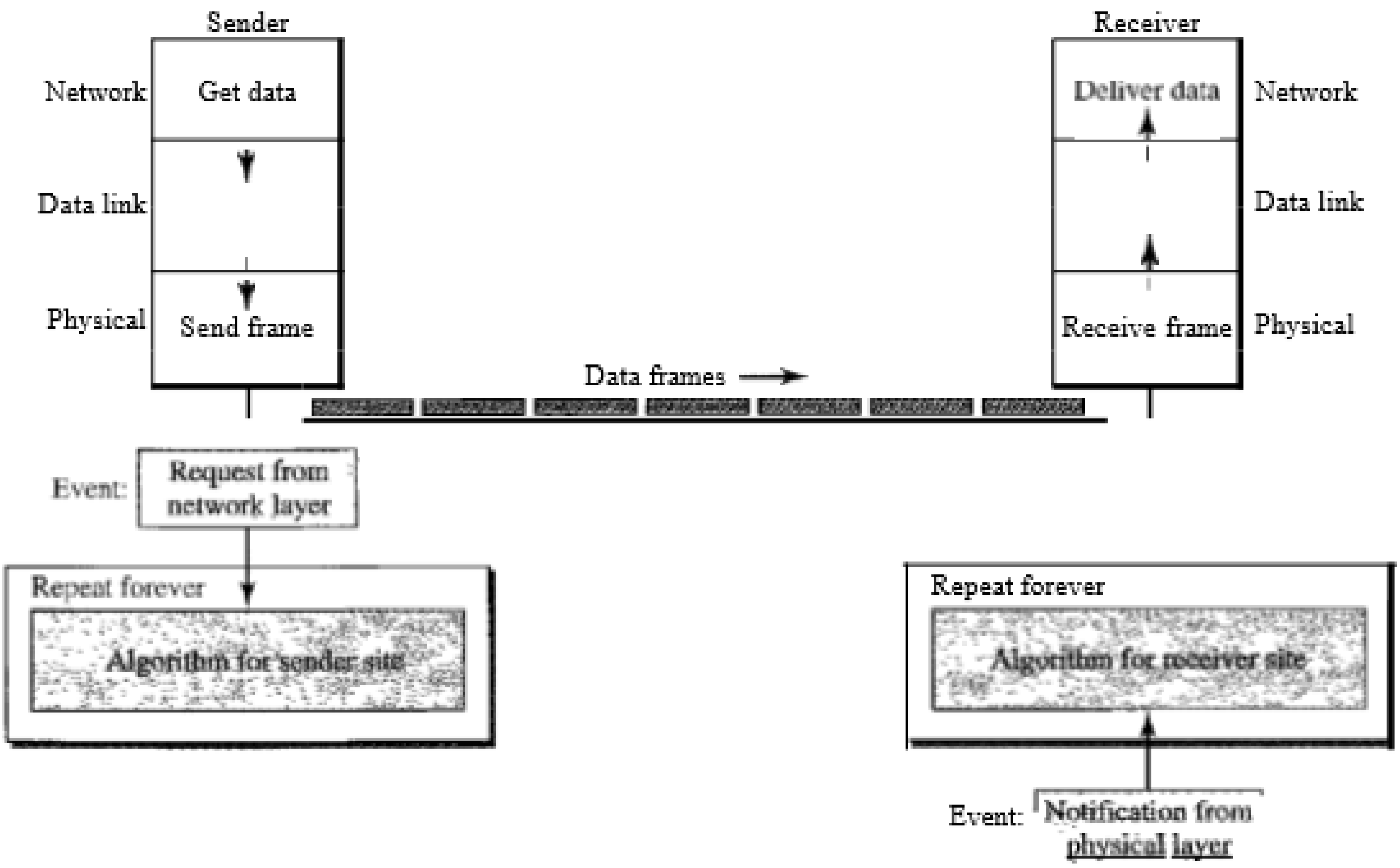


NOISELESS CHANNELS

- Let us first assume we have an ideal channel in which no frames are lost, duplicated, or corrupted.
- We introduce two protocols for this type of channel. The first is a protocol that does not use flow control; the second is the one that does. Of course, neither has error control because we have assumed that the channel is a perfect noiseless channel.

Simplest Protocol

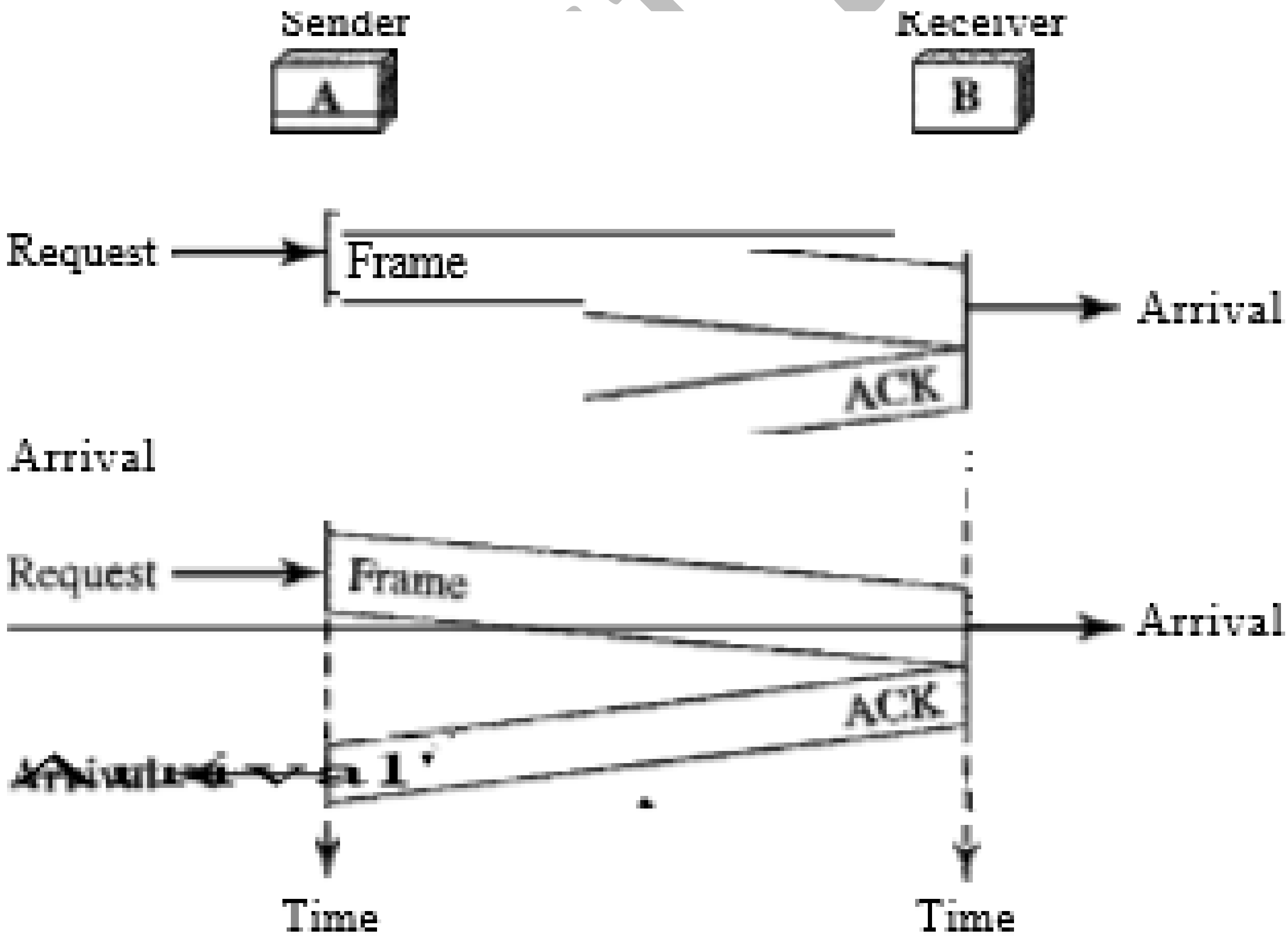
- Our first protocol, is one that has no flow or error control. it is a unidirectional protocol in which data frames are traveling in only one direction-from the sender to receiver.
- We assume that the receiver can immediately handle any frame it receives with a processing time that is small enough to be negligible.
- The data link layer of the receiver immediately removes the header from the frame and hands the data packet to its network layer, which can also accept the packet immediately. In other words, the receiver can never be overwhelmed with incoming frames. There is no need for flow control in this scheme.
- We need to elaborate on the procedure used by both data link layers. The sender site cannot send a frame until its network layer has a data packet to send. The receiver site cannot deliver a data packet to its network layer until a frame arrives.



Sender
Network
Get data
Data link
Physical
Send frame
Receiver
Deliver data
Network
Data link
Receive frame
Physical
Data frames
Event:
Request from network layer
Repeat forever
Algorithm for sender site
Repeat forever
Algorithm for receiver site
Event:
Notification from physical layer

Stop-and-Wait Protocol

- If data frames arrive at the receiver site faster than they can be processed, the frames must be stored until their use.
- Normally, the receiver does not have enough storage space, especially if it is receiving data from many sources. This may result in either the discarding of frames or denial of service.
- To prevent the receiver from becoming overwhelmed with frames, we somehow need to tell the sender to slow down. There must be feedback from the receiver to the sender.
- The protocol we discuss now is called the Stop-and-Wait Protocol because the sender sends one frame, stops until it receives confirmation from the receiver (okay to go ahead), and then sends the next frame.
- We still have unidirectional communication for data frames, but auxiliary ACK frames (simple tokens of acknowledgment) travel from the other direction. We add flow control to our previous protocol.



NOISY CHANNELS

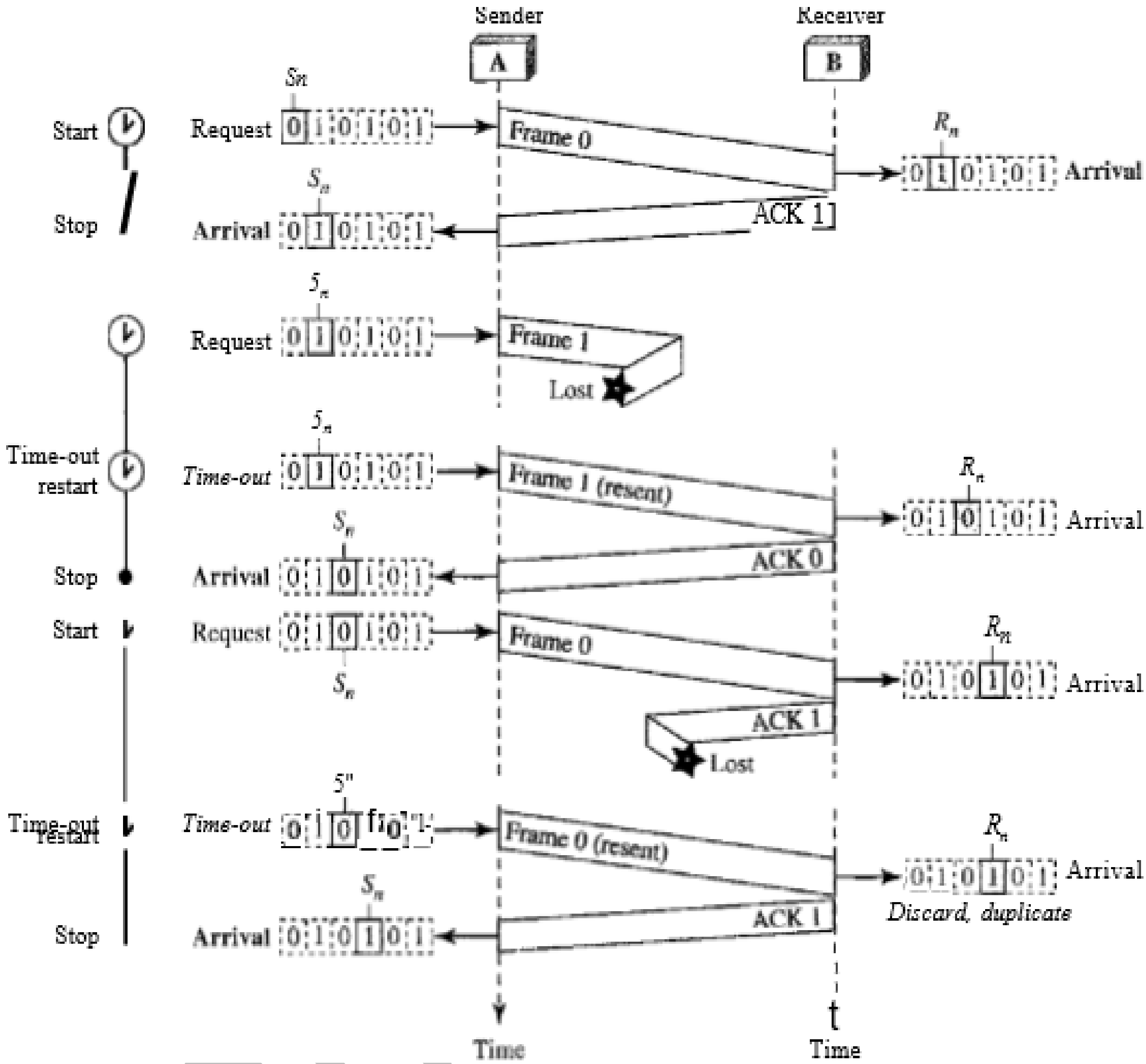
- Although the Stop-and-Wait Protocol gives us an idea of how to add flow control to its predecessor, noiseless channels are non-existent. We can ignore the error (as we sometimes do), or we need to add error control to our protocols. We discuss three protocols in this section that use error control.

Stop-and-Wait Automatic Repeat Request

- Our first protocol, called the Stop-and-Wait Automatic Repeat Request (Stop-and Wait ARQ), adds a simple error control mechanism to the Stop-and-Wait Protocol.
- Let us see how this protocol detects and corrects errors. To detect and correct corrupted frames, we need to add redundancy bits to our data frame.
- When the frame arrives at the receiver site, it is checked and if it is corrupted, it is silently discarded. The detection of errors in this protocol is manifested by the silence of the receiver.
- Lost frames are more difficult to handle than corrupted ones. In our previous protocols, there was no way to identify a frame. The received frame could be the correct one, or a duplicate, or a frame out of order.
- The solution is to number the frames. When the receiver receives a data frame that is out of order, this means that frames were either lost or duplicated.
- The corrupted and lost frames need to be resent in this protocol. If the receiver does not respond when there is an error, how can the sender know which frame to resend?
- To remedy this problem, the sender keeps a copy of the sent frame. At the same time, it starts a timer. If the timer expires and there is no ACK for the sent frame, the frame is resent, the copy is held, and the timer is restarted.
- Since the protocol uses the stop-and-wait mechanism, there is only one specific frame that needs an ACK even though several copies of the same frame can be in the network. Since an ACK frame can also be corrupted and lost, it too needs redundancy bits and a sequence number. The ACK frame for this protocol has a sequence number field. In this protocol, the sender simply discards a corrupted ACK frame or ignores an out-of-order one.

Sequence Numbers

- As we discussed, the protocol specifies that frames need to be numbered. This is done by using sequence numbers.
- A field is added to the data frame to hold the sequence number of that frame. One important consideration is the range of the sequence numbers. Since we want to minimize the frame size, we look for the smallest range that provides unambiguous communication.
- The sequence numbers of course can wrap around. For example, if we decide that the field is m bits long, the sequence numbers start from 0, go to $2^m - 1$, and then are repeated.
- Let us reason out the range of sequence numbers we need. Assume we have used x as a sequence number; we only need to use x + 1 after that. There is no need for x + 2. To show this, assume that the sender has sent the frame numbered x. Three things can happen
- The frame arrives safe and sound at the receiver site; the receiver sends an acknowledgment. The acknowledgment arrives at the sender site, causing the sender to send the next frame numbered x + 1.
- The frame arrives safe and sound at the receiver site; the receiver sends an acknowledgment, but the acknowledgment is corrupted or lost. The sender resends the frame (numbered x) after the time-out. Note that the frame here is a duplicate. The receiver can recognize this fact because it expects frame x + I but frame x was received.
- The frame is corrupted or never arrives at the receiver site; the sender resends the frame (numbered x) after the time-out.
- We can see that there is a need for sequence numbers x and x + I because the receiver needs to distinguish between case 1 and case 2. But there is no need for a frame to be numbered x + 2.
- In case 1, the frame can be numbered x again because frames x and x + 1 are acknowledged and there is no ambiguity at either site. In cases 2 and 3, the new frame is x + I, not x + 2. If only x and x + 1 are needed, we can let x =0 and x + I == 1. This means that the sequence is 0, I, 0, I, 0, and so on.



Efficiency

- The Stop-and-Wait ARQ is very inefficient if our channel is thick and long. By thick, we mean that our channel has a large bandwidth; by long, we mean the round-trip delay is long.
- The product of these two is called the bandwidth delay product. We can think of the channel as a pipe. The bandwidth-delay product then is the volume of the pipe in bits. The pipe is always there.
- If we do not use it, we are inefficient. The bandwidth-delay product is a measure of the number of bits we can send out of our system while waiting for news from the receiver.

Transmission Delay (TT): A sender needs to put the bits in a packet on the line one by one. If the first bit of the packet is put on the line at time t1 and the last bit is put on the line at time t2, transmission delay of the packet is (t2 − t1).
T_t = (Packet length (L)) / (Transmission rate or Bandwidth (B)) = L / B

Example: Fast Ethernet LAN with the transmission rate of 100 million bits per second and a packet of 10,000 bits, the transmission delay will be:

(10,000) / (100,000,000) = 100 microseconds.

Propagation Delay: Propagation delay is the time it takes for a bit to travel from point A to point B in the transmission media.
T_P = (Distance) / (Propagation speed)

Processing Delay - It is the time required for a destination host to receive a packet from its input port, remove the header, perform an error detection procedure, and deliver the packet to the output port or deliver the packet to the upper-layer protocol (in the case of the destination host).
Delay_{pr} = Time required to process a packet in a destination host

Queuing Delay It is measured as the time a packet waits in the input queue and output queue of a router.
Delay_{qu} = The time a packet waits in input and output queues in a router

Total delay = $(n + 1)\,(T_t + T_p + \text{Delay}_{pr}) + (n)\,(\text{Delay}_{qu})$

Note:

- If we have n routers, we have $(n + 1)$ links. Therefore, we have $(n + 1)$ transmission delays related to n routers and the source, $(n + 1)$ propagation delays related to $(n + 1)$ links, $(n + 1)$ processing delays related to n routers and the destination, and only n queuing delays related to n routers.
- In most of the numerical we will consider the processing and queuing delays as 0.

Measuring Performance for Stop and Wait

- The total time is measured as:
- Total time = $T_{t\,(data)} + T_{p\,(data)} + Dealy_{que} + Delay_{pro} + T_{t\,(ack)} + T_{p\,(ack)}$
- Thus, *Total time =* $T_t + (2 * T_p)$
- Here we have taken $T_{t\,(ack)}$ as negligible as the ack size is generally very less, the T_p for data and ack are almost going to be same and queuing delay and processing delays as already discussed are kept 0.
- In general, *2 * T_p time is also called Round Trip Time (RTT)*

- Efficiency is measured as: η = Useful Time / Total Cycle time = T_t / T_t + 2 * T_p, Here, Useful time in the entire cycle time is T_t and for the rest 2 * T_p time we are waiting for the processing, whereas instead of waiting we could have sent more packets.
- Dividing numerator and denominator with T_t, we get: η = 1 / 1 + (2 * T_p/T_t)
- So, η = 1 / 1 + 2a, (where a = T_p / T_t)

- Effective Bandwidth / Throughput / Bandwidth Utilization is calculated as: Throughput = L / T_t + 2*T_p
- Or, dividing and multiplying the numerator with B we get
- Throughput = (L/B) * B / T_t + 2*T_p => T_t * B / T_t + 2T_p => η * B (efficiency * bandwidth)

Example: A sender uses the Stop-and-Wait ARQ protocol for reliable transmission of frames. Frames are of size 1000 bytes and the transmission rate at the sender is 80 Kbps (1Kbps = 1000 bits/second). Size of an acknowledgement is 100 bytes and the transmission rate at the receiver is 8 Kbps. The one-way propagation delay is 100 milliseconds. Assuming no frame is lost, the sender throughput is __________ bytes/second. **(Gate-2016) (2 Marks)**
Ans: 2500

Q Suppose that the stop-and-wait protocol is used on a link with a bit rate of 64 kilobits per second and 20 milliseconds propagation delay. Assume that the transmission time for the acknowledgment and the processing time at nodes are negligible. Then the minimum frame size in bytes to achieve a link utilization of at least 50% is _________. (Gate-2015) (2 Marks)

(A) 160 **(B)** 320 **(C)** 640 **(D)** 220

Answer: (B)

Q A link has a transmission speed of 10^6 bits/sec. It uses data packets of size 1000 bytes each. Assume that the acknowledgment has negligible transmission delay, and that its propagation delay is the same as the data propagation delay. Also assume that the processing delays at nodes are negligible. The efficiency of the stop-and-wait protocol in this setup is exactly 25%. The value of the one-way propagation delay (in milliseconds) is ___________. **(Gate-2015) (1 Marks)**

Answer: (12)

Q On a wireless link, the probability of packet error is 0.2. A stop-and-wait protocol is used to transfer data across the link. The channel condition is assumed to be independent from transmission to transmission. What is the average number of transmission attempts required to transfer 100 packets? **(Gate-2006) (2 Marks)**

(A) 100 **(B)** 125 **(C)** 150 **(D)** 200

Answer: (B)

Q A channel has a bit rate of 4 kbps and one-way propagation delay of 20 ms. The channel uses stop and wait protocol. The transmission time of the acknowledgement frame is negligible. To get a channel efficiency of at least 50%, the minimum frame size should be **(Gate-2005) (2 Marks)**

(A) 80 bytes **(B)** 80 bits **(C)** 160 bytes **(D)** 160 bits

Answer: (D)

Q The values of parameters for the Stop-and-Wait ARQ protocol are as given below:

- Bit rate of the transmission channel = 1 Mbps.
- Propagation delay from sender to receiver = 0.75 ms.
- Time to process a frame = 0.25 ms.
- Number of bytes in the information frame = 1980.
- Number of bytes in the acknowledge frame = 20.
- Number of overhead bytes in the information frame = 20.

Assume there are no transmission errors. Then, the transmission efficiency (expressed in percentage) of the Stop-and-Wait ARQ protocol for the above parameters is _____________ (correct to 2 decimal places). **(Gate-2017) (2 Marks)**

Example: Consider a stop and wait protocol which sends 10 packets from source to destination out of which every 4th packet is lost, then how many packets are we going to send in total.

Consider that the packets transmitted are

1 2 3 4 5 6 7 8 9 10

Now counting the retransmissions of every 4th packet we get

1 2 3 4 4(Ret) 5 6 7 7(Ret) 8 9 10 10(Ret)

1st Loss (at 4), 2nd Loss (at 7), 3rd Loss (at 10)

Total Transmissions = 13 transmissions.

- **The total transmissions of stop and wait protocol if the percentage of packet loss is *p*, and the total packet sent are *n*: n (1 / 1 - p).**

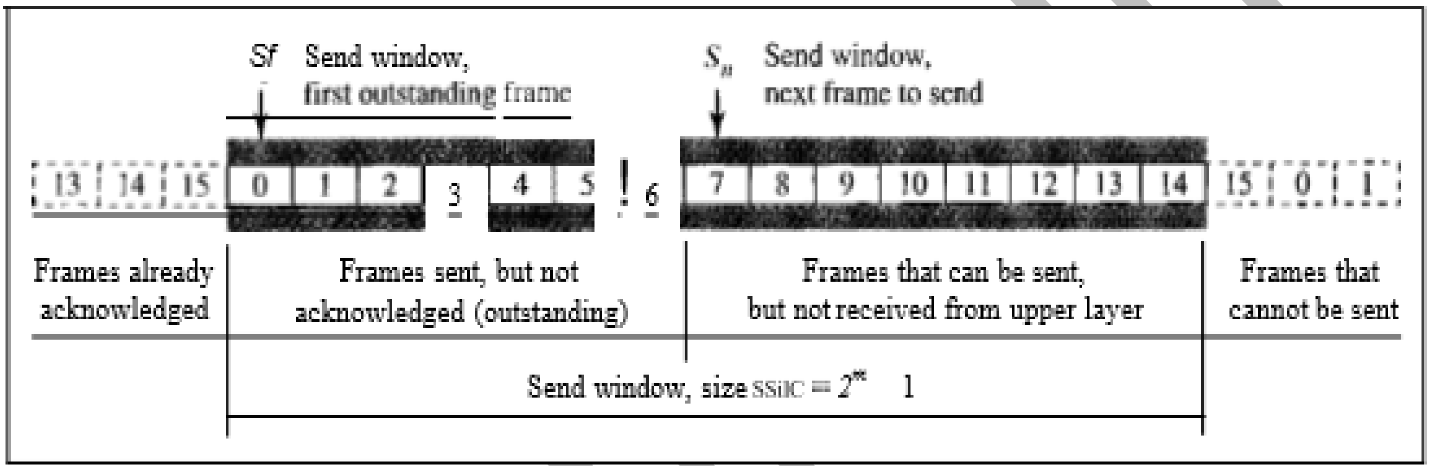
Go-Back-N Automatic Repeat Request

- To improve the efficiency of transmission (filling the pipe), multiple frames must be in transition while waiting for acknowledgment.
- In other words, we need to let more than one frame be outstanding to keep the channel busy while the sender is waiting for acknowledgment. The first is called Go-Back-N Automatic Repeat Request.
- In this protocol we can send several frames before receiving acknowledgments; we keep a copy of these frames until the acknowledgments arrive.

Sliding Window

- In this protocol (and the next), the sliding window is an abstract concept that defines the range of sequence numbers that is the concern of the sender and receiver.
- In other words, the sender and receiver need to deal with only part of the possible sequence numbers. The range which is the concern of the sender is called the send sliding window; the range that is the concern of the receiver is called the receive sliding window.
- We discuss both here. The send window is an imaginary box covering the sequence numbers of the data frames which can be in transit. In each window position, some of these sequence numbers define the frames that have been sent; others define those that can be sent.
- The maximum size of the window is $2^m - 1$.
- The window at any time divides the possible sequence numbers into four regions. The first region, from the far left to the left wall of the window, defines the sequence numbers belonging to frames that are already acknowledged.
- The sender does not worry about these frames and keeps no copies of them. The second region, colored, defines the range of sequence numbers belonging to the frames that are sent and have an unknown status.
- The sender needs to wait to find out if these frames have been received or were lost. We call these outstanding frames.
- The third range, defines the range of sequence numbers for frames that can be sent; however, the corresponding data packets have not yet been received from the network layer.
- Finally, the fourth region defines sequence numbers that cannot be used until the window slides. The window itself is an abstraction; three variables define its size and location at any time.

- We call these variables Sf (send window, the first outstanding frame), Sn (send window, the next frame to be sent), and Ssize (send window, size).
- The variable Sf defines the sequence number of the first (oldest) outstanding frame. The variable Sn holds the sequence number that will be assigned to the next frame to be sent. Finally, the variable Ssize defines the size of the window, which is fixed in our protocol.
- The receive window makes sure that the correct data frames are received and that the correct acknowledgments are sent. The size of the receive window is always I. The receiver is always looking for the arrival of a specific frame. Any frame arriving out of order is discarded and needs to be resent.



Timers

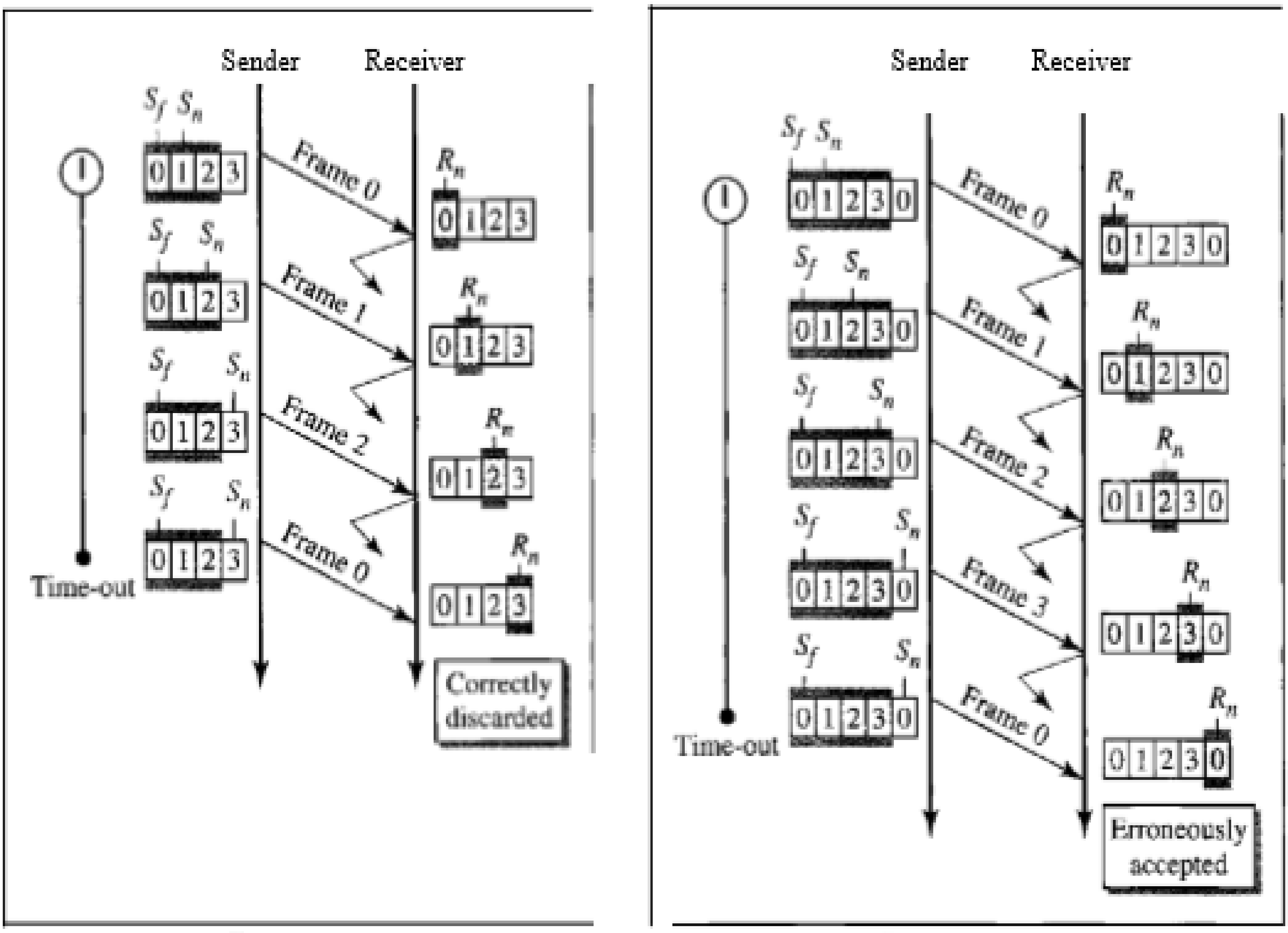
- Although there can be a timer for each frame that is sent, in our protocol we use only one. The reason is that the timer for the first outstanding frame always expires first; we send all outstanding frames when this timer expires.

Acknowledgment

- The receiver sends a positive acknowledgment if a frame has arrived safe and 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.
- The silence of the receiver causes the timer of the unacknowledged frame at the sender site to expire. This, in turn, causes the sender to go back and resend all frames, beginning with the one with the expired timer.
- The receiver does not have to acknowledge each frame received. It can send one cumulative acknowledgment for several frames.

Resending a Frame

- When the timer expires, the sender resends all outstanding frames. For example, suppose the sender has already sent frame 6, but the timer for frame 3 expires. This means that frame 3 has not been acknowledged; the sender goes back and sends frames 3, 4,5, and 6 again. That is why the protocol is called Go-Back-NARQ.



a. Window size $< 2^m$

b. Window size $= 2^m$

Q Consider a network connecting two systems located 8000 kilometres apart. The bandwidth of the network is 500×10^6 bits per second. The propagation speed of the media is 4×10^6 meters per second. It is needed to design a Go-Back-N sliding window protocol for this network. The average packet size is 10^7 bits. The network is to be used to its full capacity. Assume that processing delays at nodes are negligible. Then, the minimum size in bits of the sequence number field has to be ________. **(Gate-2015) (2 Marks)**

Answer: (8)

Q A 1Mbps satellite link connects two ground stations. The altitude of the satellite is 36,504 km and speed of the signal is 3×10^8 m/s. What should be the packet size for a channel utilization of 25% for a satellite link using go-back-127 sliding window protocol? Assume that the acknowledgment packets are negligible in size and that there are no errors during communication. **(Gate-2008) (2 Marks)**

(A) 120 bytes **(B)** 60 bytes **(C)** 240 bytes **(D)** 90 bytes

Answer: (A)

Q Station A needs to send a message consisting of 9 packets to Station B using a sliding window (window size 3) and go-back-n error control strategy. All packets are ready and immediately available for transmission. If every 5th packet that A transmits gets lost (but no acks from B ever get lost), then what is the number of packets that A will transmit for sending the message to B? **(Gate-2006) (2 Marks)**

(A) 12 **(B)** 14 **(C)** 16 **(D)** 18

Answer: (C)

Q A 20 Kbps satellite link has a propagation delay of 400 ms. The transmitter employs the "go back n ARQ" scheme with n set to 10. Assuming that each frame is 100 bytes long, what is the maximum data rate possible? **(Gate-2004) (2 Marks)**

(A) 5Kbps **(B)** 10Kbps **(C)** 15Kbps **(D)** 20Kbps

Answer: (B)

Example: Consider that the transmission delay for a link is 1ms and Propagation delay is 49.5 sec. What should be the maximum window size in order to achieve maximum efficiency? And minimum number of bits required in sequence field?

The maximum window size = 1 + 2a = (1 + 2 (49.5 / 1)) = 100

Minimum number of bits in sequence number field = $\log_2(100)$ = 6.7 = 7 bits

In some cases the sequence number bits are often reserved, suppose in the above example we say that Sequence number bits are fixed to be 5.

Then, with 5 bits we can generate only 2^5 = 32 sequence numbers, i.e. only 32 frames can be sent in a pipeline.

Thus efficiency is = 32 / 100 = 32%

Thus window size can also be defined in a much better way as: *Min (1 + 2a, 2^N)*

Sliding window protocol is implemented through: *Go-Back N ARQ and Selective Repeat ARQ*

Example: If T_t is given as 1msec and T_p is given as 99.5 ms and the protocol used is GB-10, calculate the throughput is bandwidth is given as 40 MBPS?

Since, we know to achieve maximum efficiency the W_s = 1 + 2a = 1 + 199 = 200

But we have already fixed the W_s = 10.

So efficiency = 10 / 200 = 1 / 20 = 5%.

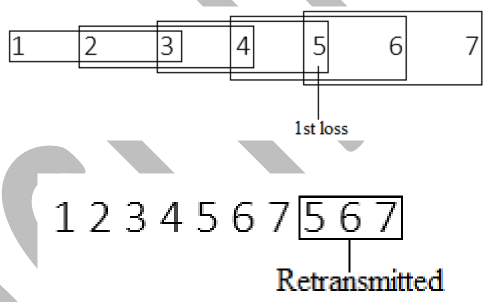
Throughput = Efficiency * Bandwidth = (5 / 100) * 40 MBPS = 2 MBPS

Example: In GB-3 if every 5th packet is lost, if we need to send 10 packets then how many transmissions are required?

Let the packets be: 1 2 3 4 5 6 7 5 6 7 8 9 10

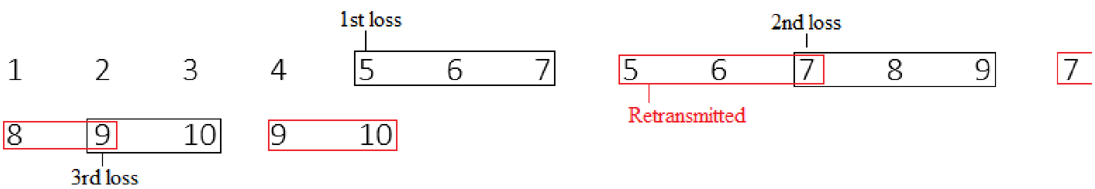
The first packet lost will be at 5, and W_s = 3.

The entire window will now be retransmitted, i.e. 5, 6, 7 will be retransmitted.



Now, we start again counting from first 6, then we get to know that next loss will be at frame 7. So again the entire frame i.e. 7 8 9 is retransmitted.

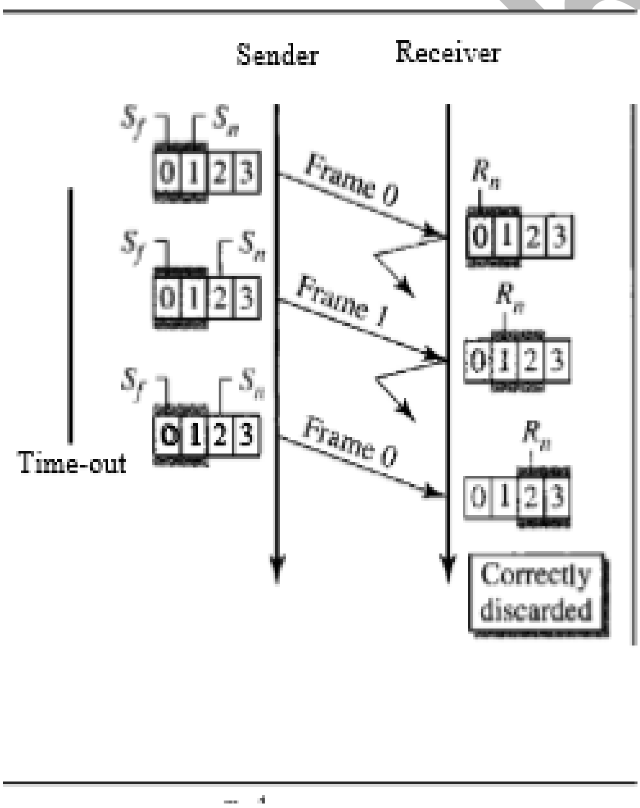
Thus, following the same procedure we get;

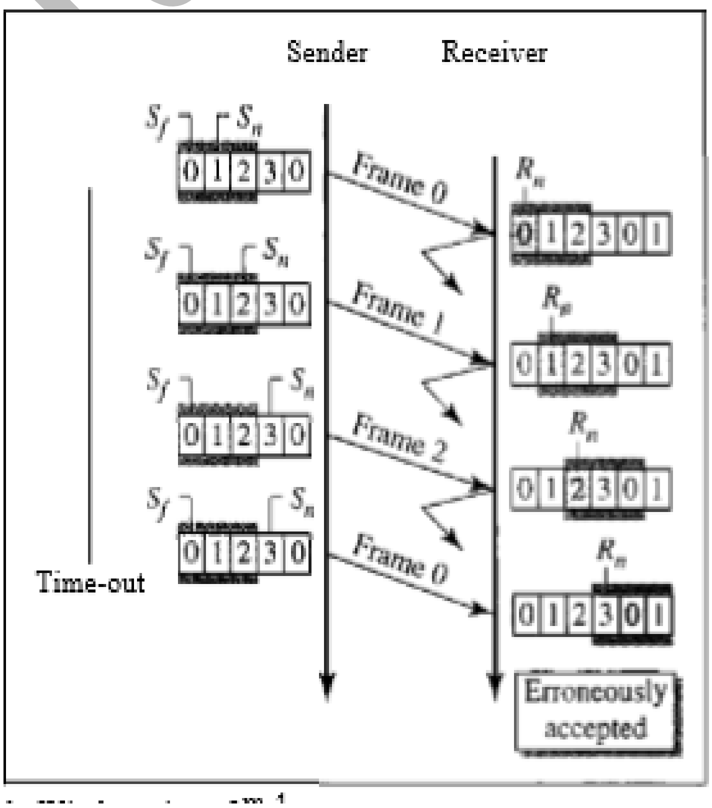


Total Transmissions: 18

Selective Repeat Automatic Repeat Request

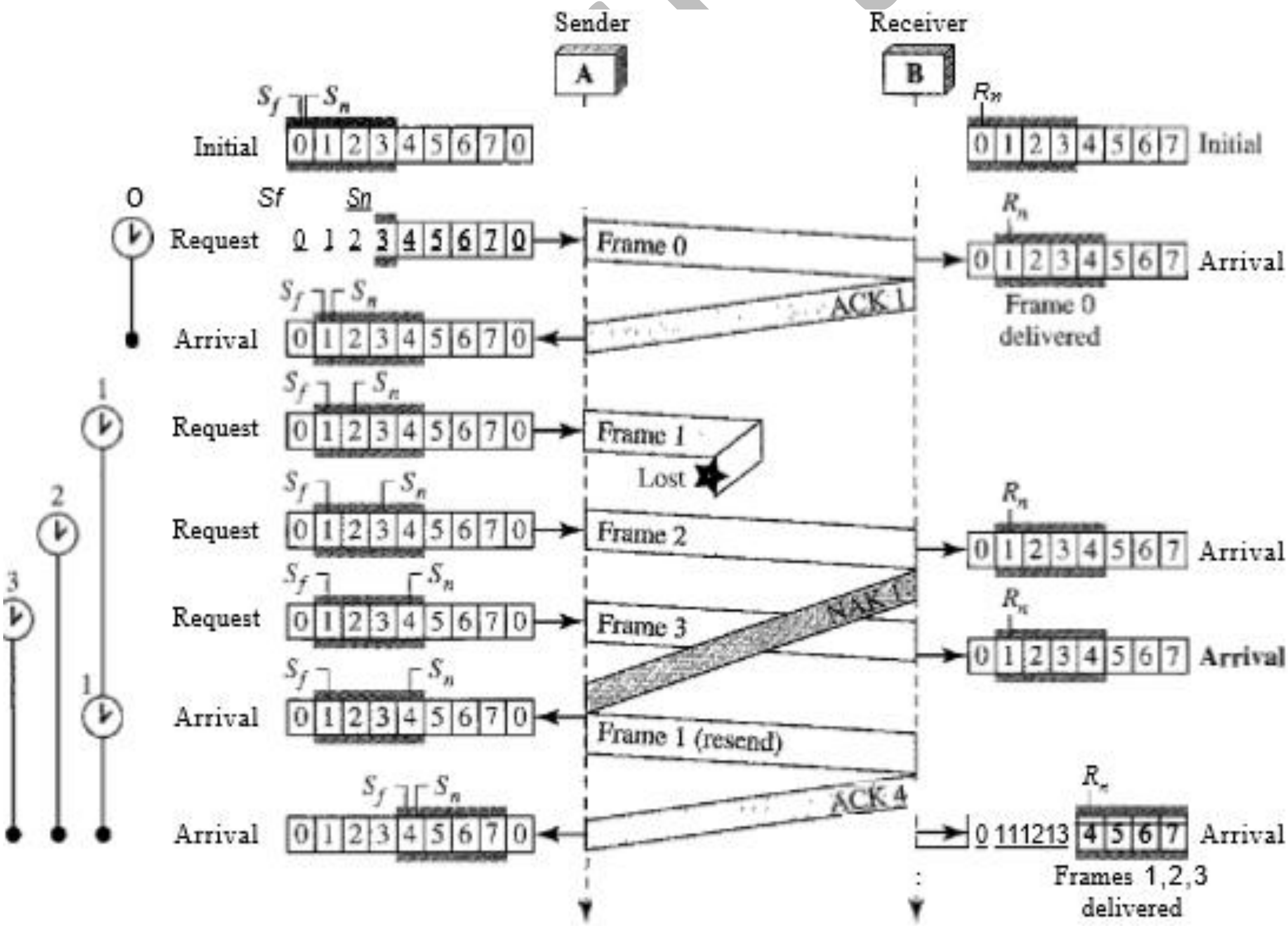
- Go-Back-N ARQ simplifies the process at the receiver site. The receiver keeps track of only one variable, and there is no need to buffer out-of-order frames; they are simply discarded.
- However, this protocol is very inefficient for a noisy link. In a noisy link a frame has a higher probability of damage, which means the resending of multiple frames. This resending uses up the bandwidth and slows down the transmission.
- For noisy links, there is another mechanism that does not resend N frames when just one frame is damaged; only the damaged frame is resent. This mechanism is called Selective Repeat ARQ.
- It is more efficient for noisy links, but the processing at the receiver is more complex. The Selective Repeat Protocol allows as many frames as the size of the receive window to arrive out of order and be kept until there is a set of in-order frames to be delivered to the network layer.
- Because the sizes of the send window and receive window are the same, all the frames in the send frame can arrive out of order and be stored until they can be delivered.





- In Selective Repeat ARQ, the size of the sender and receiver window must be at most one-halfof2^m

- The handling of the request event is similar to that of the previous protocol except that one timer is started for each frame sent. The arrival event is more complicated here. An ACK or a NAK frame may arrive. If a valid NAK frame arrives, we just resend the corresponding frame. If a valid ACK arrives, we use a loop to purge the buffers, stop the corresponding timer. and move the left wall of the window. The time-out event is simpler here; only the frame which times out is resent.
- Analysis Here we need more initialization. In order not to overwhelm the other side with NAKs, we use a variable called Nak Sent. To know when we need to send an ACK, we use a variable called Ack Needed.
- Both of these are initialized to false. We also use a set of variables to mark the slots in the receive window once the corresponding frame has arrived and is stored.
- If we receive a corrupted frame and a NAK has not yet been sent, we send a NAK to tell the other site that we have not received the frame we expected.
- If the frame is not corrupted and the sequence number is in the window, we store the frame and mark the slot. If contiguous frames, starting from Rn have been marked, we deliver their data to the network layer and slide the window.



Piggybacking

- The three protocols we discussed in this section are all unidirectional: data frames flow in only one direction although control information such as ACK and NAK frames can travel in the other direction.
- In real life, data frames are normally flowing in both directions: from node A to node B and from node B to node A. 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 frame is carrying data from A to B, it can also carry control information about arrived (or lost) frames from B; when a frame is carrying data from B to A, it can also carry control information about the arrived (or lost) frames from A.
- Note that each node now has two windows: one send window and one receive window. Both also need to use a timer. Both are involved in three types of events: request, arrival, and time-out.
- However, the arrival event here is complicated; when a frame arrives, the site needs to handle control information as well as the frame itself. Both of these concerns must be taken care of in one event, the arrival event. The request event uses only the send window at each site; the arrival event needs to use both windows. An important point about piggybacking is that both sites must use the same algorithm. This algorithm is complicated because it needs to combine two arrival events into one. We leave this task as an exercise.

Sequence and Acknowledgement Numbers

To improve the efficiency of transmission (to fill the pipe), multiple packets must be in transition while the sender is waiting for acknowledgment

- In order to maximize the efficiency, the window size (W_s) = (1+ 2a)
- The minimum number of sequence numbers required = (1 + 2a)
- Number of bits required for sequence numbers = ceil ($\log_2$ (1 + 2a))

Example: Consider an SR protocol, with sender window size equals to 3 and we have to send 10 packets and every 5th packet is lost.

Soln: Let the packets be:

1 2 3 4 **5** 6 7 8 9 10

We know that the first packet loss will be at 5, instead of transmitting the entire window we will only transfer packet 5.

1 2 3 4 5 5 6 7 8 9 10

Next packet loss will be at 9, as it is the next fifth packet.

1 2 3 4 5 5 6 7 8 9 9 10

Total transmissions: 12

Comparison

	Stop and Wait ARQ	Go back N	Selective Repeat	Remarks
Efficiency	1 / (1+2a)	N / (1+2a)	N / (1+2a)	Go back N and Selective Repeat gives better efficiency than Stop and Wait ARQ.
Window Size	Sender Window Size = 1 Receiver Window Size = 1	Sender Window Size = N Receiver Window Size = 1	Sender Window Size = N Receiver Window Size = N	Buffer requirement in Selective Repeat is very large. If the system does not have lots of memory, then it is better to choose Go back N.
Minimum number of sequence numbers required	2	N+1	2 x N	Selective Repeat requires large number of bits in sequence number field.

Retransmissions required if a packet is lost	Only the lost packet is retransmitted	The entire window is retransmitted	Only the lost packet is retransmitted	Selective Repeat is far better than Go back N in terms of retransmissions required.
Bandwidth Requirement	Bandwidth requirement is Low	Bandwidth requirement is high because even if a single packet is lost, entire window has to be retransmitted. Thus, if error rate is high, it wastes a lot of bandwidth.	Bandwidth requirement is moderate	Selective Repeat is better than Go back N in terms of bandwidth requirement.
CPU usage	Low	Moderate	High due to searching and sorting required at sender and receiver side	Go back N is better than Selective Repeat in terms of CPU usage.
Level of difficulty in Implementation	Low	Moderate	Complex as it requires extra logic and sorting and searching	Go back N is better than Selective Repeat in terms of implementation difficulty.
Acknowledgements	Uses independent acknowledgement for each packet	Uses cumulative acknowledgements (but may use independent acknowledgements as well)	Uses independent acknowledgement for each packet	Sending cumulative acknowledgements reduces the traffic in the network but if it is lost, then the ACKs for all the corresponding packets are lost.
Type of Transmission	Half duplex	Full duplex	Full duplex	Go back N and Selective Repeat are better in terms of channel usage.

Q Consider two hosts X and Y, connected by a single direct link of rate 10^6 bits/sec. The distance between the two hosts is 10,000 km and the propagation speed along the link is 2 x 10^8 m/s. Hosts X send a file of 50,000 bytes as one large message to hosts Y continuously. Let the transmission and propagation delays be p milliseconds and q milliseconds, respectively. Then the vales of p and q are: **(Gate-2017) (2 Marks)**

(A) p = 50 and q = 100 **(B)** p = 50 and q = 400

(C) p = 100 and q = 50 **(D)** p = 400 and q = 50

Ans: d

Q Consider a 128 × 10^3 bits / second satellite communication link with one way propagation delay of 150 milliseconds. Selective retransmission (repeat) protocol is used on this link to send data with a frame size of 1 kilobyte. Neglect the transmission time of acknowledgement. The minimum number of bits required for the sequence number field to achieve 100% utilization is ___________. **(Gate-2016) (2 Marks)**
ANSWER 4

Q Consider a selective repeat sliding window protocol that uses a frame size of 1 KB to send data on a 1.5 Mbps link with a one-way latency of 50 msec. To achieve a link utilization of 60%, the minimum number of bits required to represent the sequence number field is _________. **(Gate-2014) (2 Marks)**
ANSWER 5

Q Consider a source computer(S) transmitting a file of size 10^6 bits to a destination computer(D) over a network of two routers (R_1 and R_2) and three links(L_1, L_2, and L_3). L_1connects S to R_1; L_2 connects R_1 to R_2; and L_3 connects R_2 to D. Let each link be of length 100 km. Assume signals travel over each link at a speed of 10^8 meters per second. Assume that the link bandwidth on each link is 1Mbps. Let the file be broken down into 1000 packets each of size 1000 bits. Find the total sum of transmission and propagation delays in transmitting the file from S to D? **(Gate-2012) (2 Marks)**

(A) 1005 ms **(B)** 1010 ms **(C)** 3000 ms **(D)** 3003 ms

Answer: (A)

Q Frames of 1000 bits are sent over a 10^6 bps duplex link between two hosts. The propagation time is 25ms. Frames are to be transmitted into this link to maximally pack them in transit (within the link).

Let I be the minimum number of bits that will be required to represent the sequence numbers distinctly assuming that no time gap needs to be given between transmission of two frames.

a) i = 2 b) i = 3 c) i = 4 d) i = 5

Q Suppose that the sliding window protocol is used with the sender window size of 2^i, where i is the numbers of bits as mentioned earlier and acknowledgements are always piggy backed. After sending 2^i frames, what is the minimum time the sender will have to wait before starting transmission of the next frame? (Identify the closest choice ignoring the frame processing time) **(Gate-2009) (2 Marks)**

a) 16ms **b)** 18ms **c)** 20ms **d)** 22ms

Ans: b

Q The distance between two stations *M* and *N* is *L* kilometres. All frames are *K* bits long. The propagation delay per kilometre is *t* seconds. Let *R* bits/second be the channel capacity. Assuming that processing delay is negligible, the *minimum* number of bits for the sequence number field in a frame for maximum utilization, when the *sliding window protocol* is used, is: **(Gate-2007) (2 Marks)**

a) $\lceil \log_2(2LtR+2K/K) \rceil$ **b)** $\lceil \log_2(2LtR/K) \rceil$

c) $\lceil \log_2(2LtR+K/K) \rceil$ **d)** $\lceil \log_2(2LtR+K/2K) \rceil$

ANSWER C

Q Station A uses 32-byte packets to transmit messages to Station B using a sliding window protocol. The round-trip delay between A and B is 80 milliseconds and the bottleneck bandwidth on the path between A and B is 128 kbps. What is the optimal window size that A should use? **(Gate-2006) (2 Marks)**

(A) 20 **(B)** 40 **(C)** 160 **(D)** 320

Answer: (B)

Q The maximum window size for data transmission using the selective reject protocol with n-bit frame sequence numbers is: **(Gate-2005) (1 Marks)**

(A) 2^n **(B)** 2^(n-1) **(C)** 2^n – 1 **(D)** 2^(n-2)

Answer: (B)

Q In a sliding window ARQ scheme, the transmitter's window size is N and the receiver's window size is M. The minimum number of distinct sequence numbers required to ensure correct operation of the ARQ scheme is (Gate-2004) (2 Marks)

(A) min (M, N) **(B)** max (M, N) **(C)** M + N **(D)** MN

Answer: (C)

Q Host A is sending data to host B over a full duplex link. A and B are using the sliding window protocol for flow control. The send and receive window sizes are 5 packets each. Data packets (sent only from A to B) are all 1000 bytes long and the transmission time for such a packet is 50 μs. Acknowledgement packets (sent only from B to A) are very small and require negligible transmission time. The propagation delay over the link is 200 us. What is the maximum achievable throughput in this communication? **(Gate-2003) (2 Marks)**

(A) 7.69×10^6 bytes per second **(B)** 11.11×10^6 bytes per second

(C) 12.33×10^6 bytes per second **(D)** 15.00×10^6 bytes per second

Answer: (B)