Acknowledgment

The receiver sends positive acknowledgments 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 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 Frame

When a frame is damaged, the sender goes back and sends a set of frames starting from the damaged one up to the last one sent. 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, so the sender goes back and sends frames 3, 4, 5, 6 again. That is why the protocol is called Go-Back-*N* ARQ.

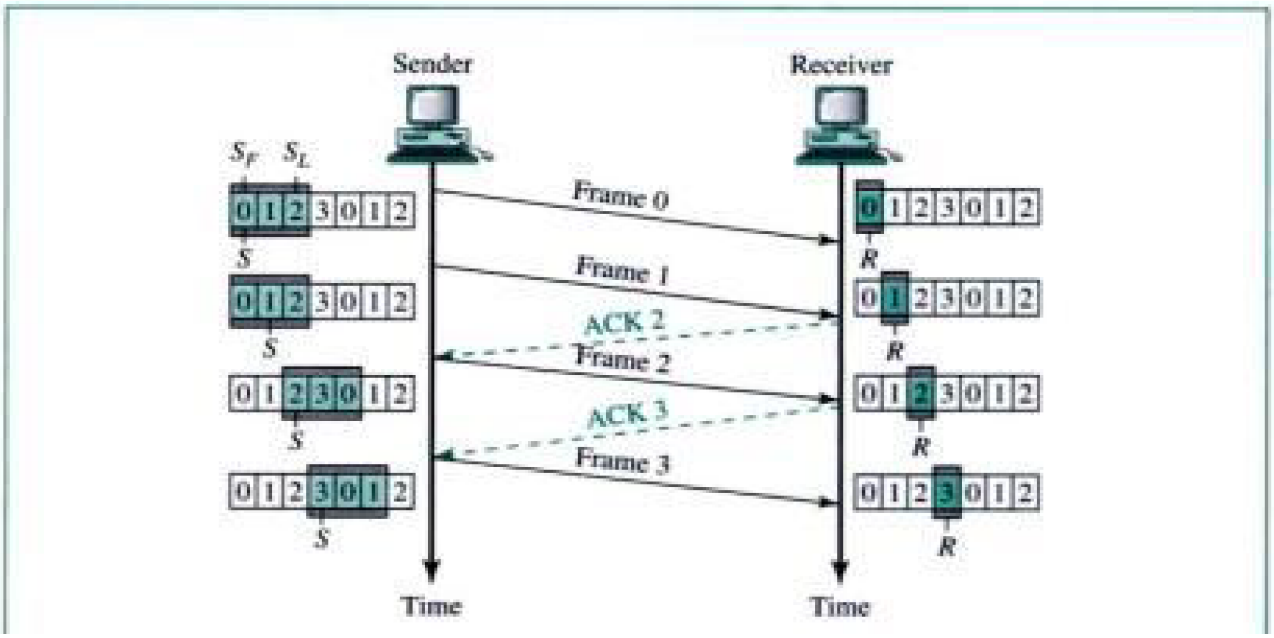
Operation

Let us see what happens in various situations.

Normal Operation

Figure 11.9 shows a normal operation of this mechanism. The sender keeps track of the outstanding frames and updates the variables and windows as the acknowledgments arrive.

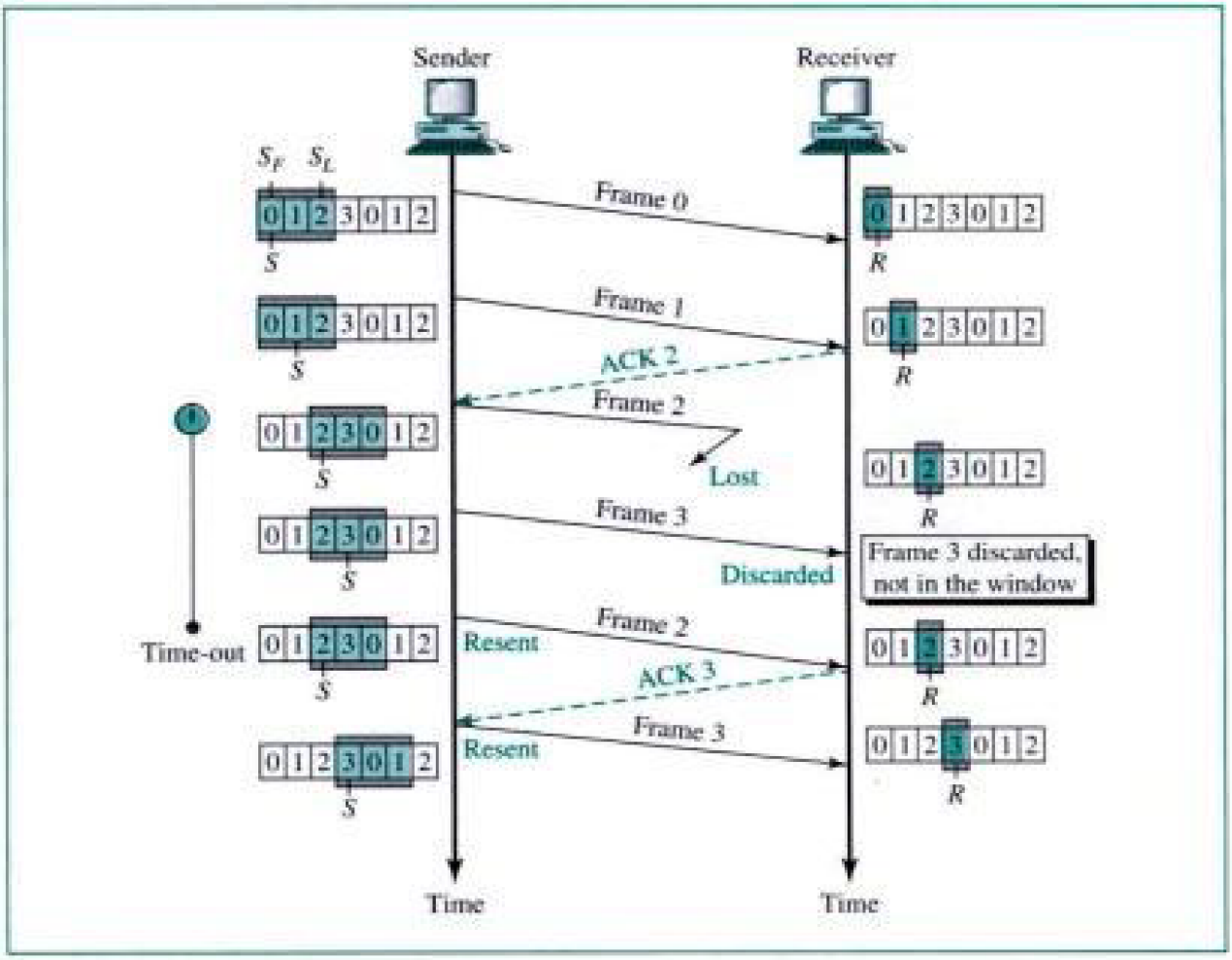
Figure 11.9 *Go-Back-N ARQ, normal operation*



Damaged or Lost Frame

Now let us see what happens if a frame is lost. Figure 11.10 shows that frame 2 is lost. Note that when the receiver receives frame 3, it is discarded because the receiver is expecting frame 2, not frame 3 (according to its window). After the timer for frame 2 expires at the sender site, the sender sends frames 2 and 3 (it goes back to 2).

Figure 11.10 *Go-Back-N ARQ, lost frame*



Damaged or Lost Acknowledgment

If an acknowledgment is damaged or lost, we can have two situations. If the next acknowledgment arrives before the expiration of any timer, there is no need for retransmission of frames because acknowledgments are cumulative in this protocol. ACK 4 means ACK 1 to ACK 4. So if ACK 1, ACK 2, and ACK 3 are lost, ACK 4 covers them. However, if the next ACK arrives after the time-out, the frame and all the frames after that are resent. Note that the receiver never resends an ACK. We leave the figure and details as an exercise.

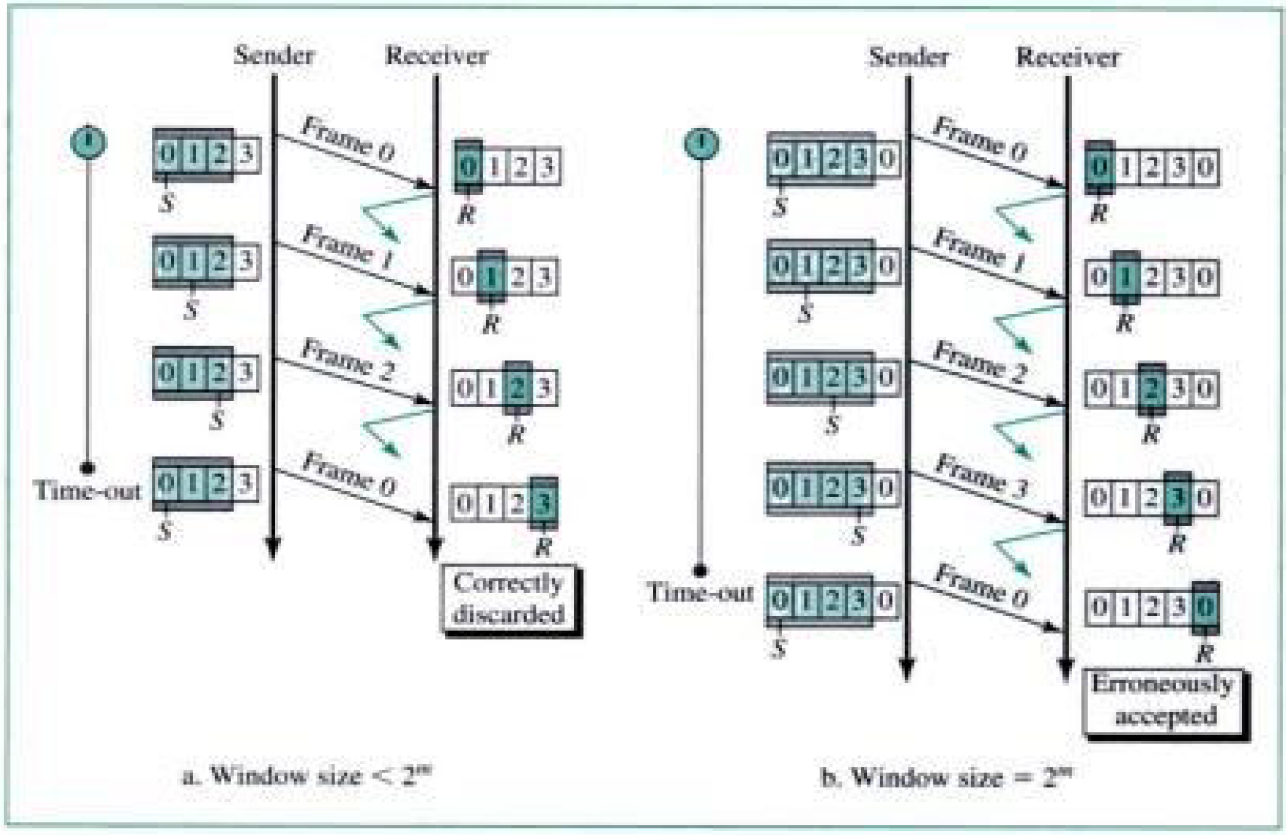
Delayed Acknowledgment

A delayed acknowledgment also triggers the resending of frames. Again, we leave the

Sender Window Size

We can now show why the size of the sender window must be less than 2^m. As an example, we choose $m = 2$, which means the size of the window can be $2^m - 1$, or 3. Figure 11.11 compares a window size of 3 and 4.

Figure 11.11 *Go-Back-N ARQ: sender window size*



If the size of the window is 3 (less than 2^2) and all three acknowledgments are lost, the frame 0 timer expires and all three frames are resent. However, the window of the receiver is now expecting frame 3, not frame 0, so the duplicate frame is correctly discarded. On the other hand, if the size of the window is 4 (equal to 2^2) and all acknowledgments are lost, the sender will send the duplicate of frame 0. However, this time the window of the receiver expects to receive frame 0, so it accepts frame 0, not as a duplicate, but as the first frame in the next cycle. This is an error.

In Go-Back-*N* ARQ, the size of the sender window must be less than 2^m; the size of the receiver window is always 1.

Bidirectional Transmission and Piggybacking

As in the case of Stop-and-Wait ARQ, Go-Back-*N* ARQ can also be bidirectional. We can also use piggybacking to improve the efficiency of the transmission. However, note that each direction needs both a sender window and a receiver window. We leave the configuration of the windows as an exercise.

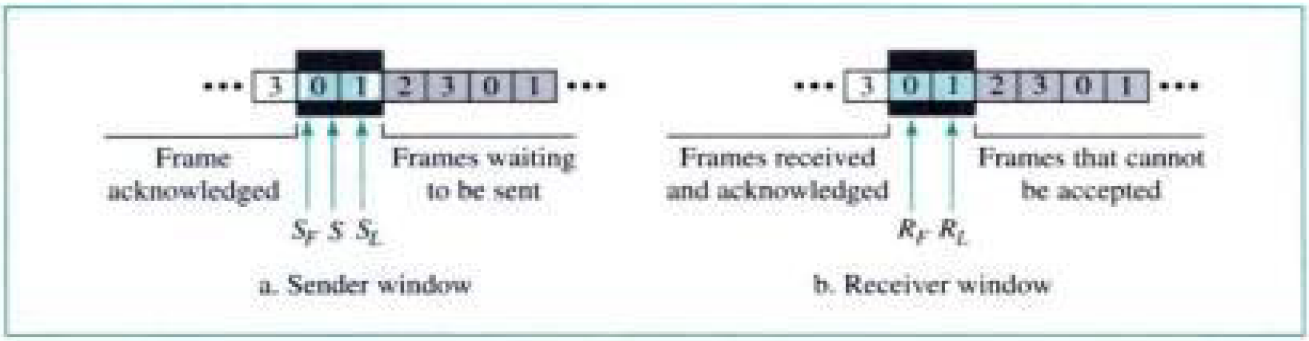
11.4 SELECTIVE REPEAT ARQ

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.

Sender and Receiver Windows

The configuration of the sender and its control variables for Selective Repeat ARQ are the same as those for Go-Back-*N* ARQ. However, for reasons to be discussed later, the size of the window should be at most one-half of the value 2^m. The receiver window size must also be this size. This window, however, specifies the range of the accepted received frame. In other words, in Go-Back-*N*, the receiver is looking for one specific sequence number; in Selective Repeat, the receiver is looking for a range of sequence numbers. The receiver has two control variables R_F and R_L to define the boundaries of the window. Figure 11.12 shows the sender and receiver windows.

Figure 11.12 *Selective Repeat ARQ, sender and receiver windows*



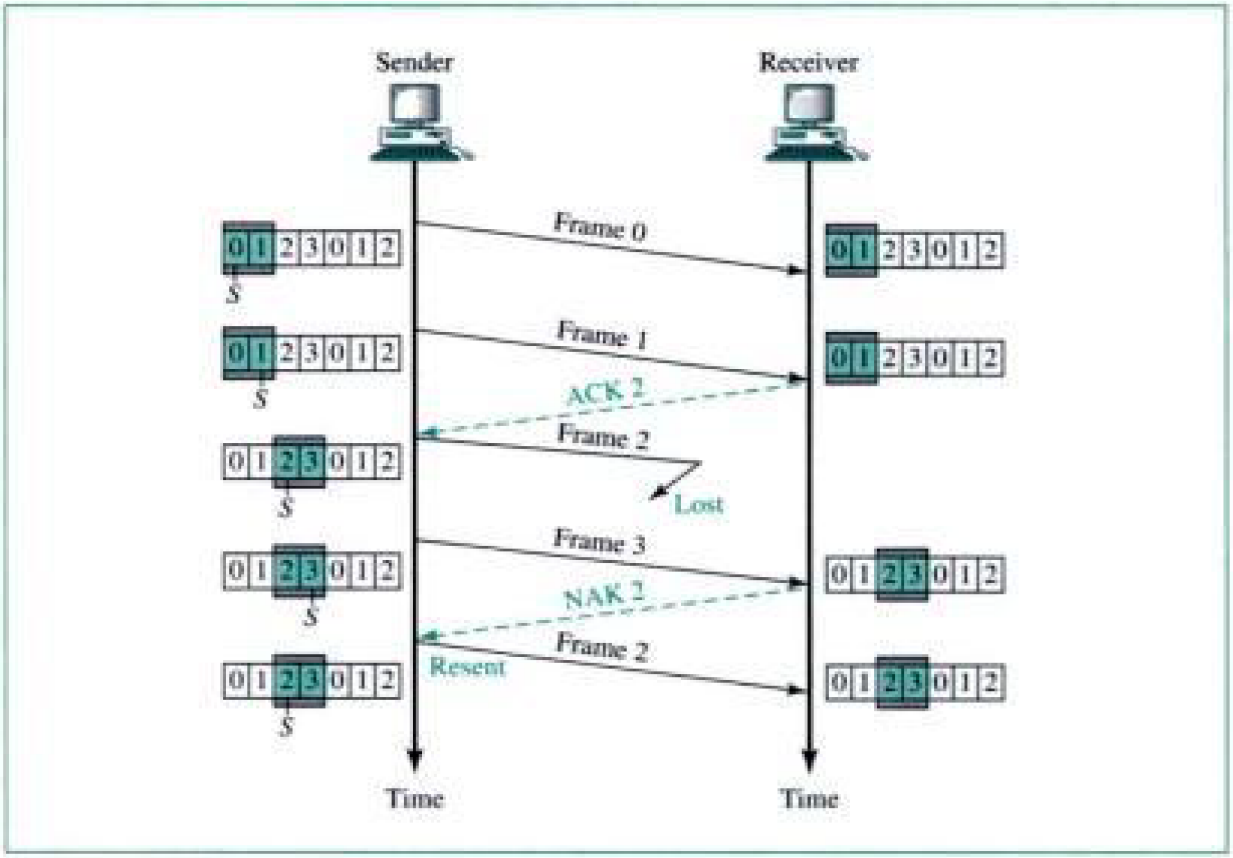
Selective Repeat ARQ also defines a **negative acknowledgment (NAK)** that reports the sequence number of a damaged frame before the timer expires.

Operation

Let us show the operation of the mechanism with an example of a lost frame, as shown in Figure 11.13.

Frames 0 and 1 are accepted when received because they are in the range specified by the receiver window. When frame 3 is received, it is also accepted for the same reason. However, the receiver sends a NAK 2 to show that frame 2 has not been received. When the sender receives the NAK 2, it resends only frame 2, which is then accepted

Figure 11.13 *Selective Repeat ARQ, lost frame*



Lost and Delayed ACKs and NAKs

We leave lost and delayed ACKs and NAKs as exercises. Note that the sender also sets a timer for each frame sent.

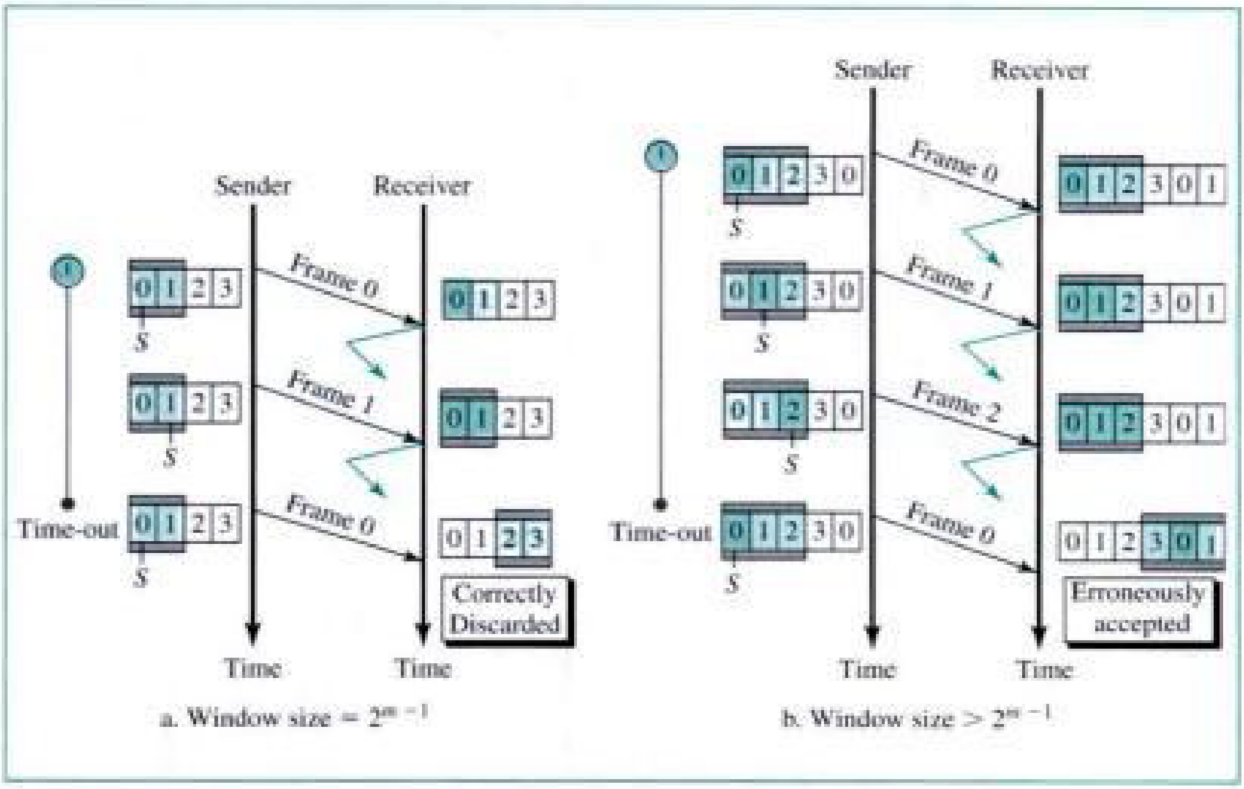
Sender Window Size

We can now show why the size of the sender and receiver windows must be at most one-half of 2^m. For an example, we choose $m = 2$, which means the size of the window should be $2^m/2$, or 2. Figure 11.14 compares a window size of 2 with a window size of 3.

If the size of the window is 2 and all acknowledgments are lost, the timer for frame 0 expires and frame 0 is resent. However, the window of the receiver is now expecting frame 2, not frame 0, so this duplicate frame is correctly discarded. When the size of the window is 3 and all acknowledgments are lost, the sender sends a duplicate of frame 0. However, this time, the window of the receiver expects to receive frame 0 (0 is part of the window), so it accepts frame 0, not as a duplicate, but as the first frame in the next cycle. This is clearly an error.

In Selective Repeat ARQ, the size of the sender and receiver window must be at most one-half of 2^m.

Figure 11.14 *Selective Repeat ARQ, sender window size*



Bidirectional Transmission and Piggybacking

As in the case of Stop-and-Wait ARQ and the Go-Back-*N* ARQ, Selective Repeat ARQ can also be bidirectional. We can use piggybacking to improve the efficiency of the transmission. However, note that each direction needs both a sender window and a receiver window. We leave the configuration of the windows as an exercise.

Bandwidth-Delay Product

A measure of the efficiency of an ARQ system is the product of the bandwidth (in bits per second) and the round-trip delay (in seconds). If the link has an adequate bandwidth, the sender will exhaust its window quickly and will wait for the acknowledgments to come. If the delay is long, the sender can also exhaust its window while waiting. So the product of these two factors can be used to define the efficiency of an ARQ system. 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.

Example 1

In a Stop-and-Wait ARQ system, the bandwidth of the line is 1 Mbps, and 1 bit takes 20 ms to make a round trip. What is the bandwidth-delay product? If the system data frames are 1000 bits in length, what is the utilization percentage of the link?

Solution

The bandwidth-delay product is

The system can send 20,000 bits during the time it takes for the data to go from the sender to the receiver and then back again. However, the system sends only 1000 bits. We can say that the link utilization is only 1000/20,000, or 5%. For this reason, for a link with high bandwidth or long delay, use of Stop-and-Wait ARQ wastes the capacity of the link.

Example 2

What is the utilization percentage of the link in Example 1 if the link uses Go-Back-*N* ARQ with a 15-frame sequence?

Solution

The bandwidth-delay product is still 20,000. The system can send up to 15 frames or 15,000 bits during a round trip. This means the utilization is 15,000/20,000, or 75 percent. Of course, if there are damaged frames, the utilization percentage is much less because frames have to be resent.

Pipelining

In networking and in other areas, a task is often begun before the previous task has ended. This is known as **pipelining.** There is no pipelining in Stop-and-Wait ARQ because we need to wait for a frame to reach the destination and be acknowledged before the next frame can be sent. However, pipelining does apply to Go-Back-*N* ARQ and Selective Repeat ARQ because several frames can be sent before we receive news about the previous frames.

Pipelining improves the efficiency of the transmission if the number of bits in transition is large with respect to the bandwidth-delay product.