ECE 333: Introduction to Communication Networks
Fall 2002

Lecture 9: Data Link Layer V

- **Sliding window ARQ protocols**
  - § **Go-back-N**
  - § **Selective repeat**

Previously we presented a basic ARQ protocol, the stop-and-wait protocol. Recall, in a stop-and-wait protocol, the sender waits until a packet is correctly acknowledged before sending the next packet. When the round trip time is large relative to the packet transmission time, this protocol has low efficiency. In this lecture, we begin discussing more efficient approaches called ***sliding window ARQ protocols***. In these protocols the sender is allowed to transmit more that one packet before receiving an acknowledgment.

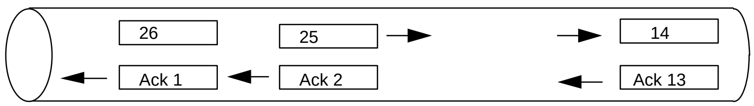
The total number of unacknowledged packets that may be sent is referred to as the sender's ***maximum window size***. For example, suppose that the sender has a window of size W. In this case, the transmitter can send packets numbered 1,2,…,W, before receiving an acknowledgment for packet 1.

Before discussing specific sliding window protocols, we consider how large the maximum window size should be.

Filling the Pipe

To maximize efficiency, the window should be chosen big enough to minimize the time the transmitter is idle. Recall in Lecture 8, we considered an example of a satellite link with a RTT of 500 msec and a packet transmission time of 20 seconds. Thus in a RTT, the transmitter can send 25 additional packets. If no errors occur, the acknowledgment from the first frame should arrive immediately after the 26th frame is sent. This allows the transmitter to send frame 27.

In this case a window size of 26 is sometimes said to ***fill the pipe***.



With no errors, this gives good efficiency, because it allows the sender to transmit continuously.

What about when errors occur?

Error Recovery in sliding window protocols

When a sliding window protocol is used, we still want to have a retransmission protocol that provides a reliable service, *i.e.* delivers packets in order, correctly, and only once. There are two classes of such protocols that are commonly used, the first is called ***go-back-n,*** and the second is called ***selective repeat***.

For both types of protocols, as with stop-and-wait, the sender numbers each packet, and the receiver returns numbered acknowledgments when a packet is correctly received. However, notice that with a sliding window protocol, as opposed to stop-and-wait, more than 1 bit will be needed for sequence numbers. Recall the sender's ***maximum window size*** is the maximum number of unacknowledged frames it is allowed to have sent at any time. We will also refer to the ***sender window*** as the set of frames that have been sent but are not yet acknowledged. Therefore, the number of frames in the sender window must be no greater than the maximum window size.

Go Back N

In a ***Go Back N*** protocol, the transmitter has a maximum window size of N packets. The receiver behaves essentially the same as in stop-and-wait. That is it keeps a counter of the current frame it is waiting to accept, and will accept only that frame. When the receiver receives any frame it returns an ACK to the transmitter containing the number of the next packet expected.

When the transmitter receives an ACK with request number RN, it can assume that every packet with number less than RN has been correctly received. As with stop-and-wait, the transmitter sets a timer for each packet that is sent. When a timer expires, the transmitter goes back and resends the first unacknowledged packet.

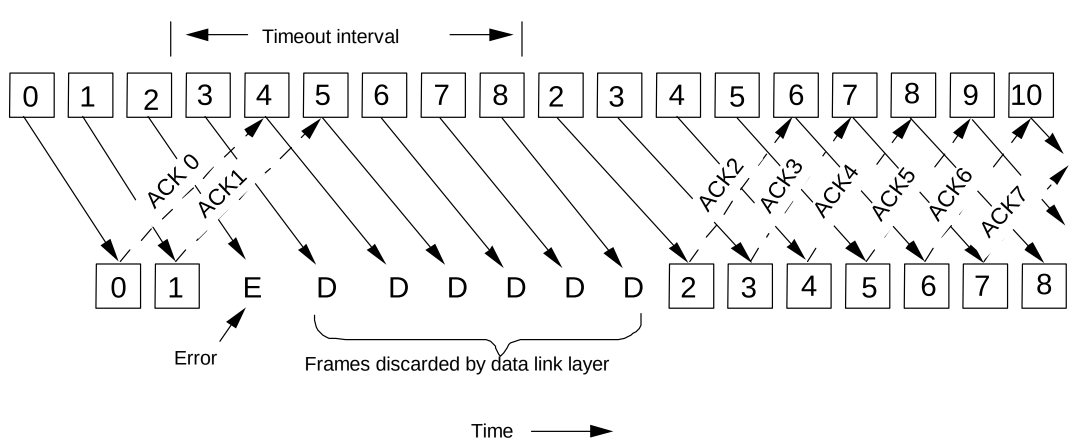
Actually, several variations of the rule used to decide when the transmitter goes back can be used. For example, the sender might go back before the timer expires if it receives two ACK's in a row containing the same request number. (Why?) The important point for the protocol to work correctly is that the transmitter eventually does go back and resends any unacknowledged frame. There are also several variations of the rule used at by the receiver to return ACK's - it may return an ACK for every received frame, or only for correct frames (in this case a timer must also be used to ensure that the receiver eventually resends an ACK if no new correct frames arrive.)

Go Back N

Pictorially, an example of the go-back-N protocol is shown below.

The sender transmits packets until it reaches the limit of the window size. If a packet times out, the sender retransmits it and goes back to send all packets after retransmitted packet.

The receiver acknowledges the frame received. (In figure, the receiver only ACKs correct frames)



Sender's Window and Sequence Numbers

In a go-back-N protocol, each packet contains a sequence number; this sequence number is denoted by a finite number of bits in the header for each packet. Suppose there are n bits available for sequence numbers in each packet. Let us consider how big the sender's maximum window size, N, can be.

Clearly it must be that $N \leq 2^n$. The question is will the protocol work correctly if $N = 2^n$?

The following example shows that the answer to this is "No".

Consider 3 bit sequence numbers and assume that N=8.
Suppose the following occurs:

- Sender sends frames 0 – 7 (using sequence numbers 0 – 7).
- ACKs for frames 0 -7 are received.
- Sender sends frames 8 – 15 (using sequence numbers 0 – 7)
- Sender receives an ACK with RN=0.

What should the sender do? (i.e., what does the RN=0 mean?)

Sender's Window and Sequence Numbers

There are two possibilities for what RN-0 could mean:

1. All 8 frames in the second group (frames 8 – 15) arrived OK at the receiver, and only the last ACK got through.

-or-

2. Frame 8 arrived at the receiver damaged, the receiver sent an RN=0, and Frames 9 – 15 are lost.

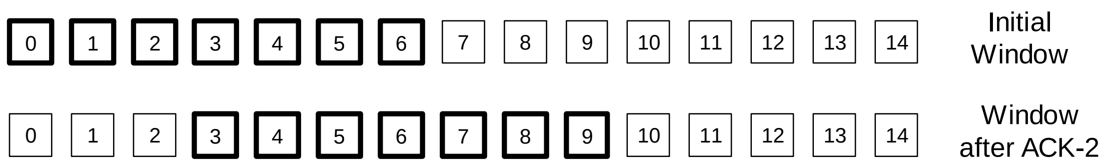
This ambiguity is clearly not a good thing. The ambiguity arises because the sender can receive an ACK to any new frame sent, and it can also receive a duplicate of the previous ACK. In this example, this makes for 9 frames that can be ACKed, and there are only 8 sequence numbers.

$\Rightarrow$ To make the Go Back N protocol work correctly, with n bits for sequence numbers, we need to restrict the maximum window size to be at most $2^n - 1$.

Go Back N Protocol

Note that the sender must buffer outstanding frames until they are acknowledged, and thus must have enough buffer space for its window size.

As frames are acknowledged, the sender can advance its window allowing additional frames to be sent, i.e., a sliding window.



Ignoring any transmission errors, by choosing a large enough window size, go-back-N can achieve near 100% efficiencies (assuming the header sizes are negligible).

Now consider the effect of errors. To recover from errors, a timer is once again used. When a packet is not acknowledged before a time-out, the transmitter will go back and begin retransmitting. If the round-trip time is known, this timer should be set to be approximately equal to the round-trip time. When the transmitter times out due to an error in a transmitted packet, all of the packets sent during the time-out time will have to be retransmitted. Even if these other packets had arrived correctly, by the operation of the protocol they would not be saved at the receiver. Such effects can limit the efficiency of go back N. How significant these effects are depends in part on the probability that a frame is in error and the number of frames sent in a round-trip time. When needed, an alternative strategy called ***selective repeat*** can be used to improve the efficiency of go back N.

Selective Repeat

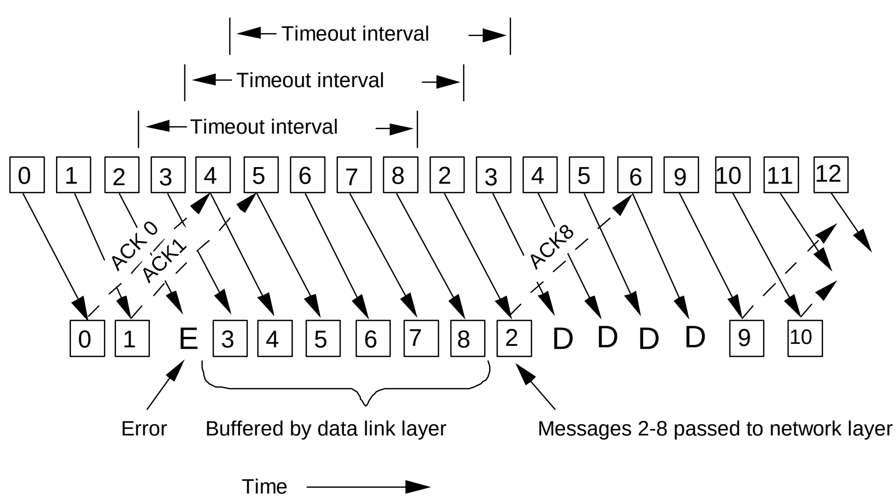
Basic idea: let receiver accept and buffer out-of-order frames that arrive correctly. Then the sender can back up when an error occurs, but will only need to resend the frames in error.

If the sender only retransmits frames in error, then the receiver is going to have to hold onto frames that were received after the one with an error, and save them until the corrupted frame is repeated. This is needed so that the receiver can pass the information to the upper layer in the correct order.

This means that in addition to the sender having a window of frames that it can transmit, the receiver now has a window of frames that it will accept. Specifically we define the ***receiver window*** at each time to start with the first unacknowledged packet and include the next N-1 packets, where N is still denotes the maximum window size at the transmitter. (Note even if one of these packets has been received, we still include it in the receiver window.)

There are several variations of selective repeat. For now, assume that the transmitter still retransmits on the basis of a timeout; we will make some improvements on this assumption later. Assume that the receiver returns acknowledgements containing the sequence number of the first unacknowledged packet. In this way an acknowledgement serves as a ***cumulative acknowledgement,*** indicating all previous packets have been correctly received.

Selective Repeat



Notice that not all the frames sent in a timeout interval need to be retransmitted. (e.g. frames 7 and 8).

How Many Sequence Number Do We Need?

With Go-Back-N, we saw that if n bits are used for sequence numbers then we must have $N \le 2^n - 1$. If instead selective repeat is used, what is the maximum window size?

Note with selective repeat, the window size refers to both the transmitter's window size as well as the receiver's window size. For a given number of bits, we should not expect to use any larger window size than with Go Back N. Suppose we use the same window size. Specifically, lets again assume that we are using 3 bit sequence numbers. That gives 8 different sequence numbers. Suppose we allow a maximum window size of 7 for the sender and the receiver. The next example shows that this is not enough sequence numbers.

Example with Too Few Sequence Numbers

Consider the following scenario:

Sender: Transmits frames 0, 1, 2, 3, 4, 5, 6
Receiver: Receives frames 0, 1, 2, 3, 4, 5, 6 (all correct)
It advances its windows to allow frames 7, 0, 1, 2, 3, 4, 5.

Assume all these acknowledgments are lost!

Sender: Times out and resends, starting with frame 0
Receiver: Puts it in buffer (since it is in the window). ACK returned will be for frame 6, since 0-6 were received, and new 0 has a hole behind it in the buffer where 7 should go.
Sender: Hearing ACK for 6, knows 0-6 arrived
It then sends 7, 0, 1, 2 . . . 5
Receiver: Receives frame 7 and passes it to network. Since frame 0 is valid, passes it also and is waiting for 1

Wrong frame 0 was given to layer above !!

The Correct Modulus for the Sequence Numbers in Selective Repeat

Solution is to have the maximally advanced receiver window never overlap sequence numbers with the minimally advanced sender's window. If the maximum window size is N, this means $2N$ sequence numbers are required. In other words with n bits available for sequence numbers, we must have $N \leq 2^n / 2 = 2^{n-1}$ for correct operation of selective repeat.

NAKs

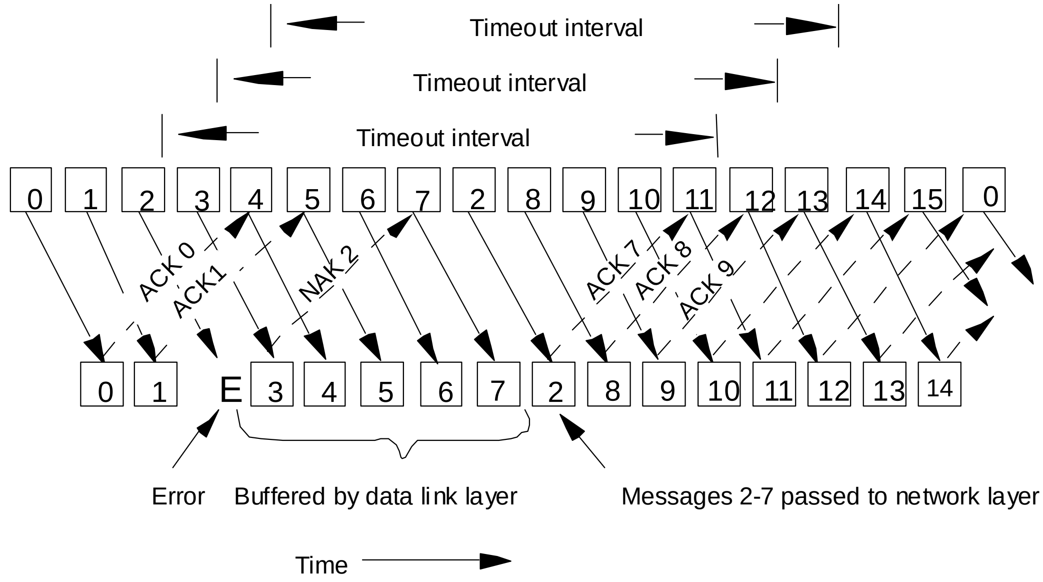
Most implementations of selective repeat allow the receiver to send a message called a NAKs (Negative Acknowledgment) when an error is detected (this may be due to a CRC failing or due to an out of sequence arrival.).

The effect of a NAK can also come from a receiving a ***duplicate acknowledgement***, i.e. two acknowledgements in a row for the same packet. Another related idea is to allow the receiver to send a selective acknowledgement, which only acknowledges a given packet, not any prior packets.

When the transmitter receives an NAK it will go back and repeat only that packet then continue sending new packets in its window. Assume there is a large window size and long time-out time. In this case, with the above approach, the sender will most likely resend only frames that are in error and not duplicate any correct frames.

An example is shown next.

Selective repeat with NAKs



Notice that with NAK's it may be advantageous to set the timeout interval much longer than the round-trip time to avoid unneeded retransmissions.

Go-Back-N vs. Selective Repeat

Conceptually, the main difference between the "Go-back N" and "Selective Repeat" protocols is the size of the receiver window. (Go-back N can be thought of as having a receiver window of size 1).

Implementation-wise, selective repeat requires more bits for the sequence numbers needed to "fill the pipe," more memory at the receiver and is more complex to implement. Selective repeat does result in higher efficiencies than go-back N; how significant these gains are depend on the error probability, packet sizes, and the round-trip time.