ECE 333: Introduction to Communication Networks
Fall 2002

Lecture 10: Data Link Layer VI

- **Performance analysis**
- **Examples**

Performance

In Lecture 8, we introduced two related performance measures for an ARQ protocol, the ***effective throughput*** and the efficiency, which corresponds to the effective throughput normalized by the channel transmission rate. We then looked at an example for these performance metrics for stop-and-wait with no errors.[1] Recall in this case the efficiency was given by

$$\eta = \frac{d}{d + h + IR}$$

where d is the number of data bits in a frame, h is the number of header bits added to each frame, R is the bit rate of the transmission channel, and I is the round-trip time. The round-trip time includes the propagation delay in each direction, plus any processing delays that are incurred. The product IR is called the ***bandwidth-delay product***; this represents the number of bits that can be sent in a round-trip time. This product arises as a fundamental parameter in many networking problems. For stop-and-wait, the larger the bandwidth-delay product, the less efficient the protocol will be for a given frame size.

Next we consider extending this analysis to the case where errors occur and to other ARQ techniques.

[1] In addition to ignoring the probability of error, w e have also made several other simplifying assumptions, such as that each frame contains the same number of bits and that the round-trip time is fixed.

Efficiency of stop-and-wait with errors

Let p_f = Prob(frame in error) and p_a = Prob(ACK in error)

Assuming all errors are independent, then

$q \equiv$Prob(Packet received correct and ACKed) = (1- p_f)(1- p_a)

Now

$$effective\,throughput = \frac{number\ of\ information\ bits/packet}{Expected\ time\ to\ send\ packet}$$

With the above model, the number of attempts until a packet is correctly received is a geometric random variable with parameter q. Thus the expected number of attempts is $1/q$. Each attempt takes $(d+h)/R + I$ seconds, thus the effective throughput is given by

$$\frac{d}{\left(\frac{(d+h)}{R}+I\right)\frac{1}{q}}$$

And the efficiency, η is:

$$\eta = \frac{d}{(d+h+IR)(1/q)} = \left(\frac{d}{d+h}\right)(1-p_f)(1-p_a)\left(\frac{1}{1+IR/(d+h)}\right)$$

The first term on the right can be identified with the loss in efficiency due to the required header, the next two terms with the loss due to errors and the final term corresponds to the loss due to errors and the ARQ technique. Note when q=1, we get our previous efficiency expression as expected.

Optimal Packet Lengths

Suppose a fixed size frame is to be used for a DLL, but you can choose the frame size. From the viewpoint of efficiency, what is the optimal frame size?

If errors are ignored, then longer frames will always improve efficiency. However when errors are taken into account, longer frames will have a higher probability of being in error (the exact relationship will depend on how errors are modeled).

Let us consider the case where bit errors are independent and occur with probability e. Suppose acknowledgements are sent in a frame that contains only a header of h bits. In this case, the efficiency can be written as:

$$\eta = \left(\frac{d}{d+h}\right)(1-e)^{h+d}(1-e)^{h}\left(\frac{1}{1+IR/(d+h)}\right)$$

This expression can be shown to be increasing with d for small d and decreasing for larger values of d. Assuming that $e << 1/(d+h)$, the optimal value of d can be shown to be approximately:

$$d_{opt} \approx \sqrt{(h+IR)/e}$$

Thus longer packet sizes should be used with smaller error probabilities and larger bandwidth-delay products. In cases, such as wireless channels, where the error probability can vary over time it can be advantageous to adjust the frame length based on the channel error rate. This idea has been used in some recent wireless standards.

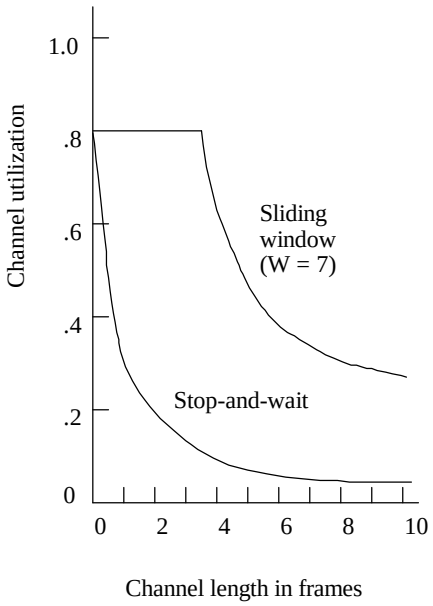
Efficiency of a sliding window protocols

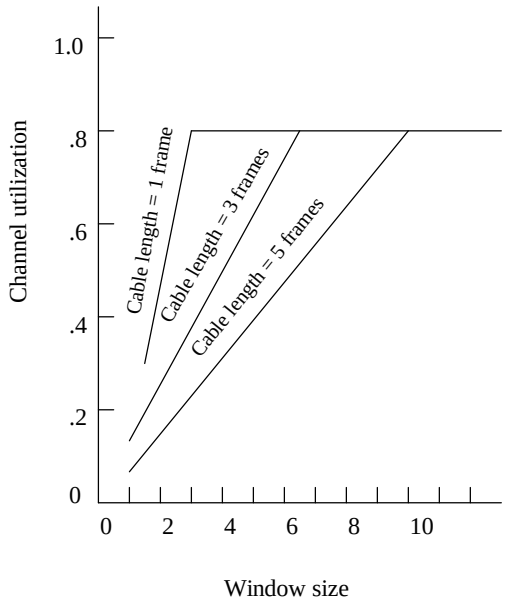
Let us consider the efficiency of sliding window protocols; first we consider the error-free case. With this assumption, both Go-back N and Selective Repeat operate the same. The efficiency of a sliding window protocol will depend on the maximum window size, N, and the bandwidth delay product. There are two distinct cases:

1. When $N < 1 + \frac{IR}{d+h}$, the transmitter will finish sending one window of frames before the first acknowledgement. In this case, Nd data bits are sent every $(d+h)/R + I$ seconds. Thus the efficiency is
$$\eta = \frac{Nd}{d+h+IR}$$
2. When $N \geq 1 + \frac{IR}{d+h}$, the transmitter can continuously transmit new frames. In this case the efficiency is
$$\eta = \frac{d}{d+h}$$

Note for a given maximum window size, the efficiency will be constant as the round-trip time increases to a point, and will then decrease after that point. This is illustrated next.

Sliding window protocols without errors





Sliding windows with errors

As with stop-and-wait, the above analysis can be extended to include packet errors.

For example consider the selective repeat protocol, where the transmitter only repeats packets that are error and then continues transmitting new packets. Assume that the window size is so large enough that the transmitter can always send new data and that the transmitter never times out. In this case, the average number of data bits successfully sent in each time interval of $(d+h)/R$ seconds is $(1-p_f)d$ where p_f still indicates the probability a packet is received correctly. The efficiency is given by

$$\eta = \left(\frac{d}{d+h}\right)\left(1-p_f\right)$$

Note we have ignored the probability that an acknowledgement was in error, why can this be done?

Next, we consider the efficiency of go-back N. As above, we assume that the window size is large enough so that the transmitter is never idle. For simplicity, we also assume that each retransmission occurs after I seconds and that ACKs are not lost.

Using go-back N, the ith packet will not be accepted by the receiver until the $(i$-1)st. We refer to the first transmission of packet i that could be accepted if it arrived correctly as the first *possibly successful transmission* of packet i. We first calculate the average time from when the transmitter begins the first possibly successful transmission of packet i until the first possibly successful transmission of packet i+1. Note that during this time, no other useful packets will be sent. If a packet is retransmitted M times during this time, then the total time is

$$\frac{d+h}{R} + (M-1)\left(I + \frac{d+h}{R}\right)$$

If each transmission is successful with probability q, then M is a geometric random variable with mean $E(M) = 1/q$. Hence the average time spent per packet is:

$$\frac{d+h}{R} + \left(\frac{1-q}{q}\right)\left(I + \frac{d+h}{R}\right).$$

The effective throughput is then given by:

$$\frac{d}{\frac{d+h}{R} + (M-1)\left(I + \frac{d+h}{R}\right)}.$$

Examples of Data Link Protocols

- HDLC (and related protocols)
- IEEE 802.2
- PPP

HDLC (High level Data Link Control)

IBM developed a DLL protocol called SDLC[2] (Synchronous Data Link Control) in the mid 1970's for use in its SNA networks (Systems Network Architecture). This was one of the first link layer protocols for providing synchronous bit-oriented communication; many other common DLL protocols are derivates of SDLC. One closely related derivative is HDLC; this protocol was standardized by the ISO and used as the DLL protocol in X.25 public networks and in other places.

HDLC has many offspring and variations. Two of the most important of these are LAPB and LAPD (Link Access Protocol B and D), both of which where standardized by the CCITT for use in Narrowband ISDN networks. Another important derivative is the IEEE 802.2 LLC protocol, which is used in IEEE 802 LANs including Ethernet.

HDLC was created at a time when it was common to have a mainframe computer connected to many terminals and has a number of features to support this type of connection.

[2] Don't worry too much about what these acronyms stand for, most people don't remember and the names are not very enlightening anyway.

HDLC Communications Modes

HDLC support a variety of "Communications Modes." The two most common are:

Normal Response Mode (NRM): In this mode one station (called the ***primary***) controls the link. The other station(s) on the link are called ***secondaries***. A secondary can only transmit after being given permission by the primary.

Asynchronous Balanced Mode (ABM): This is a combined mode, in which two stations both act as the primary and secondary. Either can initiate transmission and frames can be sent in a full-duplex manner.

HDLC Frame Format

Each HDLC frame (packet) has the format shown above. The packet header consists of 3 fields (flag, address, and control) and the trailer consists of two fields (FCS, and Flag)

The **flag** field uses the 8bit flag 01111110, and then bit stuffing is used on rest of the frame.

The **address** field can be either 1 or 2 bytes and always contains the address of the secondary. Only useful when more than one secondary is present.

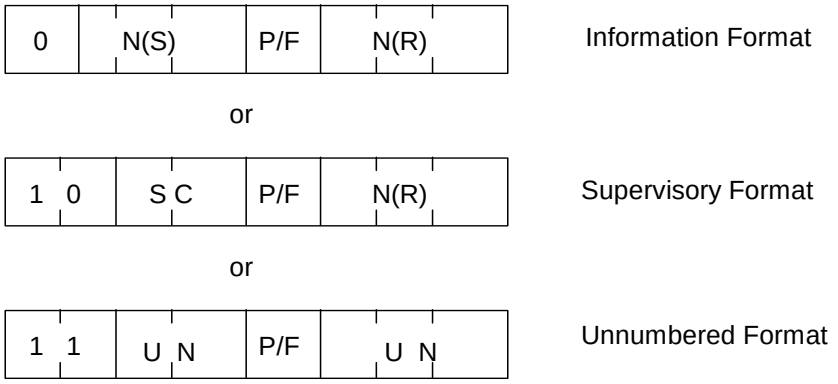
The **control** field is either 1 or 2 bytes and will be discussed below.

The **information** field is a variable length and contains the data packet from the network layer.

FSC indicates the frame check sequence; this can be either a 2 or 4 bytes and is generated using standardized CRC's.

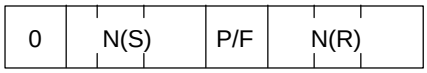
Control Field in HDLC:

Several possible formats (indicated by first 2 bits)



Control Field for HDLC Information Frames

Information frames contain user data.



N(S) = Sequence number of frame, either 3 or 7 bits.

N(R) = Acknowledgment number for piggybacked ACK, either 3 or 7 bits. (set to the next frame expected).

HDLC can be used to implement stop-and-wait, go back N and selective repeat.

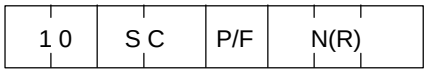
P/F = Poll/Final bit

Primary sets P = 1 to poll secondary station. This indicates that secondary can transmit.

Secondary sets F = 1 in final frame of response.

Control Fields for HDLC Supervisory Frames

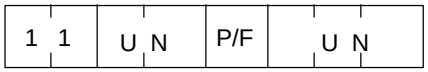
Supervisory frames are used for sending control information primarily for ARQ purposes and do not contain an information field.



The two SC bits indicate the type of supervisory frames: there are four possibilities. SC=00 indicates a **receive ready** (RR) frame. These frames are used for acknowledgements when there is no data available for piggy-backing. They can also be used to re-enable a link. SC = 01 is a **reject** (REJ) frame. REJ frames are used to send a NACK for the frame N(R) and all frames after N(R) (i.e. Go back N). SC=10 is a **receive not ready (RNR)** frame. RNR frames ACKs all frames up to N(R)-1 and indicates that the receiver can not temporarily receive any more frames (for example due to lack of memory). This is a basic example of what is called ***flow control***. Finally SC= 11 is a **selective reject (SREJ)** frame; this indicates that N(R) should be retransmitted (selective repeat). Not all of these frames are available in all variations of HDLC.

Control Fields for HDLC Unnumbered Frames

Unnumbered frames implement a variety of control functions and contain no sequence numbers - in some cases they may contain an information field.



There are 2^5 = 32 possible commands/responses, not all used.

These frames include commands for establishing a connection, choosing the mode, specifying if 3 bit or 7 bit sequence numbers are to be used, disconnecting the connection, etc.

There are also unnumbered frames for various management and testing functions.

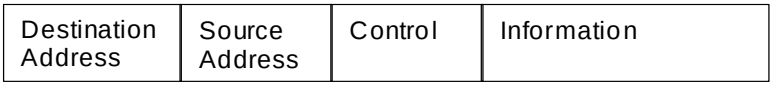
802.2 Logical Link Control

Standardized by IEEE to be the LLC sub-layer above any of the IEEE 802.x MAC protocols (such as Ethernet and Token ring). 802.2 is based on HDLC and supports 3 different types of services.

Type 1 - Unacknowledged Connectionless

Type 2 - Acknowledged Connection Oriented

Type 3 - Acknowledged Connectionless



8 bits are used for source/destination addresses; these are used to distinguish between different upper layer processes on the same host.

The Control field is modeled after HDLC it is two bytes long and has the same format as in HDLC. Information, Supervisory and Unnumbered frames are still used.

There are no checksum or frame delimiters, as these are handled by the MAC layer in 802.x LANs.

PPP - Point-to-Point Protocol

PPP is a protocol used for providing data link control over point-to-point links in the Internet. These include links connecting two routers as well as a dialup connection between a PC and an Internet Service Provide (ISP). Another protocol called SLIP (serial line IP) is also widely used, but this protocol is not an approved Internet standard and does not provide as many features as PPP. PPP is primarily defined in RFC 1661 and RFC 1662. PPP can operate over almost any physical layer that is used to provide a point-to-point transmission link.

The PPP frame format has many similarities to HDLC. It uses the same flag for framing, but uses character stuffing instead of bit stuffing, so that the resulting frame contains an integer number of bytes. The other fields are also similar to HDLC.

In addition to IP, PPP can be used to carry packets (simultaneously) from multiple network layer protocols such as DECnet, AppleTalk, OSI CLNP, etc.

PPP - Frame Format

1	1	1	1 or 2	Variable	2 or 4	1
Flag 01111110	**Address 11111111**	**Control 00000011**	**Protocol**	**Payload**	**Checksum**	**Flag 011111110**

Flag: HDLC flag - **character** stuffed if it occurs in the payload

Address: 1 byte, normally contains all 1's to indicate that all stations are to accept frame.

Control: Default value 0000 0001 which indicates an unnumbered frame (No ARQ used). For noisy links there is a numbered mode option adapted from the ABM mode of HDLC.

Protocol: Type of network layer packet in the payload. (e.g. LCP, NCP, IP, IPX, Appletalk, etc.)

Payload: Variable length, negotiated up to a maximum

Checksum: 2 or 4 (negotiated) byte standardized CRC.

LCP/NCP Packets

A protocol called the ***Link Control Protocol (LCP)*** is used to set-up and configure a link, as well as for testing and maintenance functions. For example LCP is used to specify the maximum payload size and if retransmissions are to be done. A protocol called the ***Network Control Protocol*** **(NCP)** is used configure each network layer protocol that is operated over the link. Both LCP and NCP packets are carried as payload in a PPP frames (see figure on next page). In this case the protocol field of the PPP frame is used to indicate that the payload is either a LCP or NCP packet.

Both LCP and NCP packets have the following format:

1 byte	1 byte	2 byte	variable length
Code	Identifier	Length	Options, Data, etc.

The Code field indicates the type of packet being carried (e.g. configure, disconnect, etc.)

The Identifier field is used to match requests and replies. The value used depends on the nature of the packet.

The length field indicates the total length of the packet, including the Code, Identifier, and Length fields. This is needed because LCP or NCP packets may be *padded* with extra 0's in order to make the packet length an integer number of bytes.

A PPP frame containing an LCP or NCP Packet: