LGW2ECh5solutions - 3r32r2

Cost Analysis (FPT University)

Solutions to Chapter 5

Version 1.01 (list of changes on last page)

1. Explain the difference between connectionless unacknowledged service and connectionless acknowledged service. How do the protocols that provide these services differ?

Solution:

In an acknowledged connectionless service, reliable delivery can be achieved through the use of ACK and NAK control messages. Such protocols are suited for communication over networks in which higher layers are sensitive to loss and the underlying network is inherently unreliable with a significant probability of loss or error. For example, HDLC provides for unnumbered acknowledgment service for connection setup and release.

Unacknowledged networks provide simpler and faster communication for networks that are inherently reliable or provide service to higher layers that can tolerate information loss or have built-in error recovery mechanisms.

2. Explain the difference between connection-oriented acknowledged service and connectionless acknowledged service. How do the protocols that provide these services differ?

Solution:

The use of acknowledgments can provide reliable transfer over links or networks that are prone to error, loss, and or resequencing. In a connection-oriented service, a setup phase between the sending user and receiving user establishes a context for the transfer of information. In a connection-oriented acknowledged service acknowledgments are provided to the sending user for all SDUs.

In a connectionless service, there is no prior context provided for the transfer of information between the sending user and the receiving user. The sender passes its SDU to the underlying layer without prior notice. In an acknowledged connectionless service, the sending user requires an acknowledgment of delivery of its SDU.

The protocols that provide these services are very different. Connection-oriented acknowledged service requires the use of stateful protocols that keep track of sequence numbers, acknowledgments, and timers. Connectionless services use much simpler protocols that are stateless in nature. Connectionless acknowledged service does require that the transmitting protocol track the acknowledgment of a PDU. In the simplest instance, the receiver would be required to send an ACK for correctly received PDU and the transmitter would keep a timer. If an ACK was not received in time, the transmitter would inform the user of a failure to deliver.

3. Suppose that the two end systems α and β in Figure 5.6 communicate over a connection-oriented packet network. Suppose that station α sends a 10-kilobyte message to station β and that all packets are restricted to be 1000 bytes (neglect headers); assume that each packet can be accommodated in a data link frame. For each of the links, let p be the probability that a frame incurs errors during transmission.

Solutions follow questions:

a. Suppose that the data link control just transfers frames and does not implement error control. Find the probability that the message arrives without errors at station β.

Let p be the probability that a frame incurs errors during transmission. We know the following:

Message length = 10,000 bytes

Maximum packet size = 1000 bytes
Number of packets for transmission = 10

The probability of a packet arriving error free at end system $P_{packet} = (1 - p)^3$. The probability that all packets arrive error free at end system β is $P_{error} = [(1 - p)^3]^{10} = (1 - p)^{30} \approx e^{-30p}$.

b. Suppose that error recovery is carried out end to end and that if there are any errors, the entire message is retransmitted. How many times does the message have to be retransmitted on average?

The average number of required transmissions = $1/ P_{error} = e^{30p}$.

c. Suppose that the error recovery is carried out end to end on a packet-by-packet basis. What is the total number of packet transmissions required to transfer the entire message?

The average number of transmissions per packet = $1/ P_{packet} = e^{3p}$. The total number of packet transmissions is then $10/ P_{packet} = 10e^{3p}$.

As an example suppose p = .01, then the message retransmission approach requires 1.35 message transmissions. The packet transmission approach requires 1.03 message transmissions. Clearly packet-by-packet retransmission is better.

4. Suppose that two peer-to-peer processes provide a service that involves the transfer of discrete messages. Suppose that the peer processes are allowed to exchange PDUs that have a maximum size of *M* bytes including *H* bytes of header. Suppose that a PDU is not allowed to carry information from more than one message.

Solutions follow questions:

a. Develop an approach that allows the peer processes to exchange messages of arbitrary size.

To exchange messages of arbitrary size, large messages must be segmented into parts of M-H bytes each in length to be transmitted in multiple PDUs. Small messages must be placed in a single PDU.

b. What essential control information needs to be exchanged between the peer processes?

The peer processes need to communicate information that allows for the reassembly of messages at the receiver. For example, the first PDU may contain the message length. The last PDU may contain and end-of-message marker. Sequence numbers may also be useful to detect loss in connection oriented networks and to help in reconstruction of the messages in connectionless networks. Lastly, since variable size PDUs are permitted, the size of the PDU payload must be transmitted in the PDU header.

c. Now suppose that the message transfer service provided by the peer processes is shared by several message source-destination pairs. Is additional control information required, and if so, where should it be placed?

In this case, in addition to all of the header information mentioned in b, each PDU must be labeled with a stream ID, so that the receiver can treat each stream independently when reassembling messages. This stream ID may be avoided if the source and destination operate so that they handle the transfer of a single message at a time. For example, this approach is used by AAL5 in ATM.

5. Suppose that two peer-to-peer processes provide a service that involves the transfer of a stream of bytes. Suppose that the peer processes are allowed to exchange PDUs that have a maximum size of *M* bytes, including *H* bytes of header.

Solutions follow questions:

a. Develop an approach that allows the peer processes to transfer the stream of bytes in a manner that uses the transmission line efficiently. What control information is required in each PDU?

The streams should be segmented into $M - H$ size blocks and transmitted in the PDUs. The best possible efficiency in line usage occurs when every PDU is of maximum size. If the byte stream arrives at a steady rate, full PDUs can be constructed by waiting until a sufficient number of bytes to fill a PDU payload have arrived.

The control information should include sequence numbering, payload size, and error checking. If reliability is required, the control information should also allow for ACK control messages.

b. Suppose that the bytes in the stream arrive sporadically. What is a reasonable way to balance efficiency and delay at the transmitter? What control information is required in each PDU?

If the bytes stream arrives sporadically, arriving bytes can be buffered while the transmitter waits to fill a PDU. A timer can be used to place a bound on the delay incurred waiting.

c. Suppose that the bytes arrive at a constant rate and that no byte is to be delayed by more that T seconds. Does this have an impact on the efficiency?

Yes. Because the header length is constant, larger PDUs provide more efficient data transfer. If T is very small, partially filled packets must always be sent. If T is large, larger PDUs can be transmitted so more efficient transmission can be achieved. Because the bytes arrive at a constant rate, the PDUs will all be the same length. Thus, there is no need for a PDU length field in the header.

d. Suppose that the bytes arrive at a variable rate and that no byte is to be delayed by more than T seconds. Is there a way to meet this requirement?

A timer is required that counts down from T seconds. When the first byte arrives the timer starts. When a PDU is full or when the timer expires (whichever occurs first), the PDU is transmitted and the timer is restarted upon the arrival of the next byte.

6. Suppose that two peer-to-peer processes provide a service that involves the transfer of a stream of bytes. Develop an approach that allows the stream transfer service to be shared by several pairs of users in the following cases:

Solutions follow questions:

The protocol that provides the service must be able to multiplex information from different user pairs for transfer. A basic design decision is whether the PDUs should carry information from multiple users or be constrained to carry information from a single user. The latter approach is simpler in that only a single user needs to be identified per PDU. The latter approach is complicated by the need to identify several users (actually multiplexing IDs) per PDU.

a. The bytes from each user pair arrive at the same constant rate.

If the bytes arrive at a constant rate, it may be possible to arrange a PDU structure that transfers the merged byte stream without extensive multiplexing ID information. For example, the position of a byte in the payload could identify which user it belongs to. Alternatively, bytes for each user can be buffered until they can fill a minimum sized payload. Note that the first approach will involve less delay than the second approach.

b. The bytes from the user pairs arrive sporadically and at different rates.

Mixing bytes from the different streams poses a greater challenge in this case. The bytes from different streams need to be buffered separately until they can fill a minimum sized PDU in the

approach where each PDU carries information from a single user. Alternatively, the bytes from a stream could fill a minimum-size sub-payload that could then be carried with sub-payloads from other users on shared PDUs. The choice of minimum size payload will tradeoff efficiency against delay.

7. Consider the transfer of a single real-time telephone voice signal across a packet network. Suppose that each voice sample should not be delayed by more than 20 ms.

Solutions follow questions:

a. Discuss which of the following adaptation functions are relevant to meeting the requirements of this transfer: handling of arbitrary message size; reliability and sequencing; pacing and flow control; timing; addressing; and privacy, integrity and authentication.

Message size is important because in real-time voice, samples are delayed while waiting to fill a packet. A portion of the 20 ms delay must be apportioned to the packetization delay, which in turn determines the packet payload size. The handling of arbitrary message size is not as important as long as the desired packet size for voice can be handled.

Sequencing is important because the voice samples need to be played back in the same sequence that they were generated.

Reliability is moderately important since voice transmission can tolerate a certain level of loss and error.

Pacing and flow control are not as important because the synchronous nature of the voice signal implies that the end systems will be matched in speed.

Timing, for real-time voice transfer is important because this adaptation function helps to control the jitter in the delivered signal.

Addressing is only during the connection setup phase if we assume some for of virtual circuit packet switching method. Addressing the form of multiplexing ID may be required if multiplexing is involved.

Privacy, integrity, and authentication have traditionally not been as important as the other issues discussed above. However there will surely be applications were it is essential that the voice information be kept confidential (privacy), that the information be tamper free (integrity), and that imposters be detected and deterred (authentication.)

b. Compare a hop-by-hop approach to an end-to-end approach to meeting the requirements of the voice signal.

Sequencing and timing are the most important requirements for real-time voice. These requirements are better met in a hop-by-hop approach than in an end-to-end approach because delay performance is critical. Resequencing on an end-to-end basis may lead to excessive delay. Hop-by-hop controls on transfer delay may be critical to achieving real-time transfer.

8. Suppose that a packet network is used to transfer *all* the voice signals that arrive at the base station of a cellular telephone network to a telephone office. Suppose that each voice sample should not be delayed by more than 20 ms.

Solutions follow questions:

a. Discuss which of the following adaptation functions are relevant to meeting the requirements of this transfer: handling of arbitrary message size; reliability and sequencing; pacing and flow control; timing; addressing; and privacy, integrity and authentication.

The following table summarizes parts (a) and (b):

Adaptation function	Relevant in Upstream Direction (part a)	Relevant in Downstream Direction (part b)
Handling of arbitrary message size	No	No
Reliability and sequencing	Yes	Yes
Pacing and flow control	No	No
Timing	Yes	Yes
Addressing	Yes	Yes
Privacy, integrity and authentication	Yes	No

Because traffic in a cellular network consists of constant rate streams of information, the message size need not be arbitrary, and the timing and sequence of packets is essential for service. Addressing is necessary to identify the two end callers, and authentication may only be necessary in the upstream direction, although if it is done at the base station, it need not be repeated. Pacing and flow control are not necessary because cellular networks service constant bit rate traffic and only admit calls that can be accommodated at that constant bit rate.

The key difference from the case of a single voice call (considered in Problem 7) is the multiplexing aspect of handling of multiple voice streams. The considerations in multiplexing multiple byte streams of Problem 6 apply. The separate voice streams could be carried in separate PDUs or they could be combined in the payload of the PDUs. The first approach is simpler but involves greater delay. The second approach can lead not only to lower delays but also to higher efficiency because larger payloads are possible using multiple voice streams.

Mobility is another aspect that needs to be considered. As users move from cell to cell, the network must track their movement and perform handoffs as cell boundaries are traversed. Changes in addresses and routing are involved in these handoffs.

b. Are the requirements the same in the opposite direction from the telephone office to the base station?

The set of requirements are essentially the same in both directions. However, the direction from the users upstream into the network is more difficult in general because of the need to deal with multiple incoming streams.

c. Do the answers to parts (a) and (b) change if the signals arriving at the base station include e-mail and other short messages?

If the signals include email and other short messages in addition to voice, further adaptation functions are required. The handling of arbitrary message sizes is needed, since email messages are variable in length. Alternatively, fixed small message sizes may be specified, as in Short Message Service (SMS). Pacing and flow control may now be needed depending on the nature of the messages and applications. Text email is low in bandwidth and likely would not require flow control, but very large messages, particularly those transmitted in the downstream direction to the cellular user would require flow control.

9. Suppose that streaming video information is transferred from a server to a user over a packet network.

Solutions follow questions:

a. Discuss which of the following adaptation functions are relevant to meeting the requirements of this transfer: handling of arbitrary message size; reliability and sequencing; pacing and flow control; timing; addressing; and privacy, integrity and authentication.

Streaming video involves the delivery of video information across a network. The term “streaming” usually implies a non-real-time situation where significant delay can be tolerated. The following table summarizes the results of part (a).

Adaptation function	Required	Discussion
Handling of arbitrary message size	No	A stream has no inherent required message size. However video and audio formats do have specific structures, which could be relevant to how transfer is carried out.
Reliability and sequencing	Yes	Reliability is important for video & audio quality. Sequencing is important but can be relaxed as buffering delay is increased.
Pacing and flow control	Maybe	See note below
Timing	Yes	Video has strict timing jitter requirements. Synchronization of different media is also important, but not an issue when the media are combined before packetization.
Addressing	Maybe	Required for point-to-point or point-to multipoint transfer, but not in broadcast system. Required for different media components.
Privacy, integrity and authentication	Maybe	Required for point-to-point or point-to multipoint transfer, but *may* not be required in broadcast system

Whether explicit pacing and flow control is necessary depends on the nature of the receiver. If the receiver is dedicated for receiving video, it should able to handle the incoming signal. If other applications may be received at the same time, flow control might be needed.

If the signal is constant-rate, uncompressed video, flow control is not possible. For a compressed signal, flow control may be implemented by using MPEG, which transmits video in layers that can be selectively discarded to reduce the bandwidth to the receiver.

b. Suppose that the user has basic VCR features through control messages that are transferred from the user to the server. What are the adaptation requirements for the control messages?

The following table summarizes the results of part (b).

Adaptation function	Required	Discussion
Handling of arbitrary message size	No	Control messages are short so this is not an issue.
Reliability and sequencing	Yes	Necessary to allow user to perform many operations in a short time period
Pacing and flow control	No	Control messages would not likely produce high bandwidth.
Timing	Yes	Interactive control has timing requirements.
Addressing	Yes	The control traffic must be addressed to the video server, but a generalized control protocol for multiple devices is possible.
Privacy, integrity and authentication	Yes	Authentication is important that only the users have control over the content they receive. Privacy for control information may be provided, but would likely not be a strict requirement.

10. Discuss the merits of the end-to-end vs. hop-by-hop approaches to providing a constant transfer delay for information transferred from a sending end system to a receiving end system.

Solution:

Jitter in networks has two primary sources – variation in queuing delay and variation in propagation delay. The former is caused because traffic intensity at nodes varies with time, so the buffering that is required at each node is variable. The latter exists primarily in connectionless networks and results because each packet in a flow can traverse a different path through the network to its destination.

Hop-by-hop approaches can be used to deal with queuing delay variation, and end-to-end approaches are required to deal with path-length variation. By setting up a path prior to transmission, a constant propagation delay can be ensured. To deal with queuing delay variation, the scheduler at a node can give delay sensitive traffic priority, ensuring that it's delay is kept below some maximum

amount. This can be based on header information in the packet. If the path for a flow has been previously specified, the number of nodes in the path is known, so an overall queuing delay maximum is, thus, insured.

End-to-end approaches can also provide constant transfer delay, for example, by attaching timestamps to packets obtained from a common clock, e.g. Global Positioning System, and buffering information at the receiving end until a target delivery time.

11. Consider the Stop-and-Wait protocol as described in the chapter. Suppose that the protocol is modified so that each time a frame is found in error at either the sender or receiver, the last transmitted frame is immediately resent.

Solutions follow questions:

a. Show that the protocol still operates correctly.

The protocol will operate correctly because the only difference is that frames are retransmitted sooner than otherwise. The detection of errors in an arriving frame at the receiver will cause an ACK to be sent sooner, possibly causing the transmitter to retransmit sooner. The detection of errors in an arriving frame at the transmitter will cause an immediate retransmission of the current information frame.

b. Does the state transition diagram need to be modified to describe the new operation?

The state transition diagram remains the same.

c. What is the main effect of introducing the immediate-retransmission feature?

The main effect is a speedup in the error recovery process.

12. In Stop-and-Wait ARQ why should the receiver always send an acknowledgment message each time it receives a frame with the wrong sequence number?

Solution:

The sender cannot send the next frame until it has received the ACK for the last frame so, if the receiver gets a frame with the wrong sequence it has to be a retransmission of the previous frame received. This means that the ACK was lost so the receiver has to ACK again to indicate the sender that it has received the frame.

13. Discuss the factors that should be considered in deciding whether an ARQ protocol should act on a frame in which errors are detected.

Solution:

If a frame is in error, then all of the information contained in it is unreliable. Hence any action taken as a result of receiving an erroneous frame should not use the information inside the frame. A viable option when an erroneous frame is received is to do nothing, and instead to rely on a timeout mechanism to initiate retransmission. However error recovery will be faster if we use a NACK message to prompt the sender to retransmit. The inherent tradeoff is between the bandwidth consumed by the NACK message and the faster recovery.

14. Suppose that a network layer entity requests its data link layer to set up a connection to another network layer entity. To setup a connection in a data link, the initiating data link entity sends a SETUP frame, such as SABM in Figure 5.47. Upon receiving such a frame, the receiving data link entity sends an acknowledgment frame confirming receipt of the SETUP frame. Upon receiving this acknowledgment the initiating entity can inform its network layer that the connection has been setup and is ready to transfer information. This situation provides an example of how *unnumbered acknowledgments* can arise for confirmed services.

Solutions follow questions:

a. Reexamine Figure 5.10 and Figure 5.11 with respect to error events that can take place and explain how these events are handled so that connection setup can take place reliably.

Fundamentally, the problem involves getting the receiver state to change from an idle state to a connected state. This is the same as getting the state to go from state (0,0) to state (1,1) in Figure 5.10. Therefore transmissions of the SETUP message must be accompanied by the start of an associated timer to trigger retransmissions. Also, the receiver must acknowledge every SETUP frame that it receives. The sender should ignore redundant SETUP acknowledgments and the receiver should ignore redundant SETUP frames. In order to allow multiple connections, each flow's SETUP and acknowledgment frames should be indexed. In the absence of connection indexing, the link should only handle one SETUP at a time until the SETUP is confirmed through an acknowledgment. Otherwise ambiguities in terms of what frame an ACK corresponds to will arise.

b. To terminate the connection, either data link layer can send a DISC frame that is then acknowledged by an unnumbered acknowledgment. Discuss the effect of the above error events and how they can be dealt with.

This problem is very similar to problem a. except that in the SETUP problem the actual transfer of frames cannot begin until a SETUP acknowledgment is received. In the disconnect case, the sender can stop transmitting after a certain number of retransmissions of the DISC message.

c. Suppose that an initiating station sends a SETUP frame twice but that the corresponding ACK frames are delayed a long time. Just as the ACK frames from the original transmissions are about to arrive, the initiating station gives up and sends a DISC frame followed by another SETUP frame. What goes wrong if the SETUP frame is lost?

If the first delayed ACK frame arrives after the initiating station sends the DISC frame and the second delayed ACK frame arrives after it sends the SETUP frame, the initiating station will assume the connection is established. Meanwhile the receiver receives the DISC frame and disconnects the connection and because the last SETUP frame is lost it remains in this state.

To resolve this problem the initiating station should not send out SETUP messages with DISC messages outstanding.

15. A 1 Mbyte file is to be transmitted over a 1 Mbps communication line that has a bit error rate of $p = 10^{-6}$.

Solutions follow questions:

The file length $n = 8 \times 10^6$ bits, the transmission rate $R = 1$ Mbps, and $p = 10^{-6}$.

a. What is the probability that the entire file is transmitted without errors? Note for n large and p very small, $(1-p)^n \approx e^{-np}$.

P[no error in the entire file] = $(1-p)^n \approx e^{-np}$, for $n >> 1$, $p << 1$

$= e^{-8} = 3.35 \times 10^{-4}$

We conclude that it is extremely unlikely that the file will arrive error free.

b. The file is broken up into N equal-sized blocks that are transmitted separately. What is the probability that all the blocks arrive correctly without error? Does dividing the file into blocks help?

A subblock of length n/N is received without error with probability:

P[no error in subblock] = $(1-p)^{n/N}$

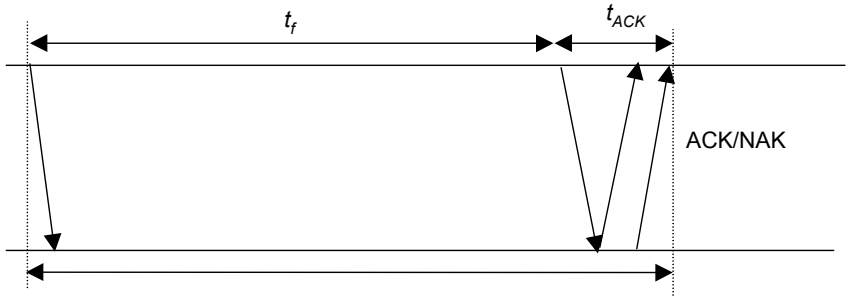
A block has no errors if all subblocks have no errors, so

$$P[\text{no error in block}] = P[\text{no errors in subblock}]^N = ((1-p)^{n/N})^N = (1-p)^n$$

So simply dividing the blocks does not help.

c. Suppose the propagation delay is negligible, explain how Stop-and-Wait ARQ can help deliver the file in error-free form. On the average how long does it take to deliver the file if the ARQ transmits the entire file each time?

Refer to the following figure for the discussion.



We assume the following:

- t_0 = basic time to send a frame and receive the ACK/NAK ≈ $t_{timeout}$
- t_{total} = total transmission time until success
- n_f = number of bits/frame
- n_a = number of bits per ACK
- n_t = number of transmissions
- P_f = probability of frame transmission error

$t_0 = t_f + t_{ACK} = n_f/R + n_a/R$ ($t_{prop} \approx 0$).

$P[n_t = i] = P[\text{one success after } i-1 \text{ failure}] = (1-P_f)P_f^{i-1}$

$t_{total} \mid i \text{ transmissions} = i.t_0$

$$E[t_{total}] = \sum_{i=1}^{\infty} it_0 P[n_t = i] = t_0(1-P_f)\sum_{i=1}^{\infty} iP_f^{i-1} = t_0(1-P_f)/(1-P_f)^2 = t_0/(1-P_f)$$

Here, $n_f = n >> n_a$ thus $t_0 \approx t_f = n/R$; and $P_f = 1 - P[\text{ no error}] = 1 - e^{-np}$

$E[\text{total}] = n/R\,(1-P_f) = n/[Re^{-np}] = 8 / (3.35 \times 10^{-4}) = 23{,}847$ seconds = 6.62 hours!

The file gets through, but only after many retransmissions.

d. Now consider breaking up the file into N blocks. (Neglect the overhead for the header and CRC bits.) On the average how long does it take to deliver the file if the ARQ transmits the blocks one at a time? Evaluate your answer for $N = 80$, 800, and 8000.

For 1 block $P_f = 1 - P_b = 1 - (1-p)^{n/N}$ and $n_f = n/N$

if $t_{prop} \approx 0$ and $n_a << n/N$: $t_0^b = n_f/R = n/NR$

$T_b = E[t_{total}^b] = t_0^b / (1 - P_f) = n(1-p)^{-n/N}/NR$ average time to transmit one block

$T = E[t_{total}] = N\,T_b = n(1-p)^{-n/N}/R = 8\,(1-p)^{-n/N} = 8\,e^{np/N}$ if $n/N >> 1$, $p << 1$

- $N = 80 \Rightarrow T \approx 8\,e^{0.1} = 8.84$ sec
- $N = 800 \Rightarrow T \approx 8\,e^{0.01} = 8.08$ sec
- $N = 8000 \Rightarrow T \approx 8\,e^{0.001} = 8.008$ sec

Each subblock has a higher probability of arriving without errors, and so requires fewer retransmissions to deliver error free. The overall delay is reduced dramatically.

e. Explain qualitatively what happens to the answer in part (d) when the overhead is taken into account.

As N increases, the effect of overhead becomes more significant because the headers constitute a bigger fraction of each subblock.

16. Consider the state transition diagram for Stop-and-Wait ARQ in Figure 5.12. Let P_f be the probability of frame error in going from station A to station B and let P_a be probability of ACK error in going from B to A. Suppose that information frames are two units long, ACK frames are one unit long, and propagation and processing delays are negligible. What is the average time that it takes to go from state (0,0) to state (0,1)? What is the average time that it then takes to go from state (0,1) to state (1,1)? What is the throughput of the system in information frames/second?

Solution:

We know that P_f is the probability of frame error and P_a is the probability of ACK error. We assume that:

- X is the random variable that represents the number of trials before a successful transmission of a frame. Each unsuccessful trial requires a timeout for retransmission. We assume that the timeout time is set to be equal to frame transmission time plus ACK transmission time.
- Y is the random variable that represents the number of trials before a successful transmission of an ACK. An ACK error will require a new successful retransmission of the frame for next ACK transmission. An ACK is not sent until a new retransmitted frame arrives at the receiver.
- X and Y follow a geometric random-variable distribution.

$T1$ = Average time to go from (0,0) to (0,1) = $(T_f + T_a)\,E(X) + T_f$

$$T1 = (2+1)\frac{P_f}{1-P_f} + 2 = 2\left[\frac{P_f}{1-P_f} + 1\right] + \frac{P_f}{1-P_f} = \frac{2}{1-P_f} + \frac{P_f}{1-P_f} = \frac{2+P_f}{1-P_f}$$

$T2$ = Average time to go from (0,1) to (1,1) = $T_1 E(Y) + T_a$

$$T2 = T_1\left[\frac{P_a}{1-P_a}\right] + T_a = \left[\frac{2+P_f}{1-P_f}\right]\left[\frac{P_a}{1-P_a}\right] + 1$$

Throughput = Frame Time / Expected Total Transmission Time = 2 / ($T1$ + $T2$)

P_f	P_a	$T1$	$T2$	Throughput
0.2	0.1	2.75	1.3055	0.4931
0.02	0.01	2.06	1.0208	0.6492
0.002	0.001	2.006	1.0020	0.6649
0	0	2.0000	1.0000	0.6667

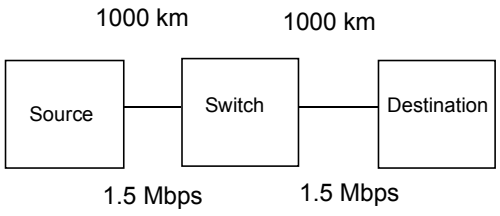
17. Write a program for the transmitter and the receiver implementing Stop-and-Wait ARQ over a data link that can introduce errors in transmission. Assume station A has an unlimited supply of frames to send to station B. Only ACK frames are sent from station B to station A. Hint: Identify each event that can take place at the transmitter and receiver and specify the required action.

Solution:

Stop and Wait Transmitter				
Current State	**Event**	**Sequence Number**	**Action**	**Next State**
Ready	Request from upper layer		Prepare frame and send; start timer	Wait
Wait	Arrival of error-free ACK frame	Correct sequence number $R_{next} = S_{last} + 1$	Increment Send Sequence Number $S_{last} = R_{next}$	Ready
Wait	Request from upper layer		New requests are blocked	Wait
Wait	Arrival of error-free ACK frame	Incorrect sequence number	Discard frame	Wait
Wait	Arrival of erroneous frame		Discard frame	Wait
Wait	Timeout Expires		Resend frame; start timer	Wait

Stop and Wait Receiver				
Current State	**Event**	**Sequence Number**	**Action**	**Next State**
Ready	Arrival of error-free frame	Expected Sequence Number $S_{last} = R_{next}$	Accept frame, increment R_{next}, $R_{next} = R_{next} + 1$, and send ACK, deliver packet to higher layer	Ready
Ready	Arrival of error-free ACK frame	Incorrect sequence number	Discard frame; send ACK with R_{next}	Ready
Ready	Arrival of erroneous frame		Discard frame	Ready

18. A 64-kilobyte message is to be transmitted from the source to the destination. The network limits packets to a maximum size of two kilobytes, and each packet has a 32-byte header. The transmission lines in the network have a bit error rate of 10^{-6}, and Stop-and-Wait ARQ is used in each transmission line. How long does it take on the average to get the message from the source to the destination? Assume that the signal propagates at a speed of 2 x 10^5 km/second.



Solution:

Message Size	65536	bytes
Max Packet Size	2048	bytes
Packet Header	32	bytes
Available for info	2016	bytes
# of packets needed	32.51	packets
Total	33	packets

bit error rate	1E-06		
bits/packet	16384		
Probability of error in packet	0.016251		1 – (1– bit_error_rate) ^ (bits/packet)
Propagation speed	2E+05	Km/s	
Distance	1000	Km	
Bandwidth	1.5	Mb/s	

We assume that the ACK error, the ACK time, and processing time are negligible.

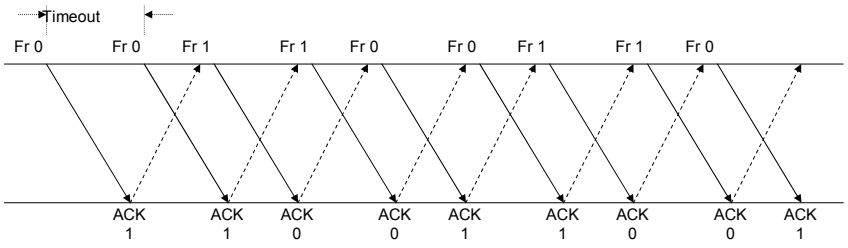
T_{prop} = distance / propagation speed = 0.0050 s
T_f = packet size / bandwidth = 0.0109 s
T_0 = T_{prop} + T_f = 0.0159 s
P_f = probability of error in packet = 0.016251

$E[T_{total}]$= T_0/ (1 - P_f) = 0.0162

There is pipelining effect that occurs as follows: After the first packet arrives at switch 1, two transmissions take place in parallel. The first packet undergoes stop-and-wait on the second link while the second packet undergoes stop-and-wait in the first link. The packet arriving at the switch cannot begin transmission on the next link until the previous packet has been delivered, so there is an interaction between the transmission times of the two packets. We will neglect this effect. The time to send every packet over two links is then the initial packet transmission time + 33 additional packet timess, and so the average time is $E[T_{total}]$ * 34 = 0.522 seconds.

19. Suppose that a Stop-and-Wait ARQ system has a time-out value that is less than the time required to receive an acknowledgment. Sketch the sequence of frame exchanges that transpire between two stations when station A sends five frames to station B and no errors occur during transmission.

Solution:



20. The Trivial File Transfer Protocol (RFC 1350) is an application layer protocol that uses the Stop-and-Wait protocol. To transfer a file from a server to a client, the server breaks the file into blocks of 512 bytes and sends these blocks to the client using Stop-and-Wait ARQ. Find the efficiency in transmitting a 1 MB file over a 10 Mbps Ethernet LAN that has a diameter of 300 meters. Assume the transmissions are error free and that each packet has 60 bytes of header attached.

Solution:

The propagation delay in an Ethernet LAN is negligible compared to the total transmission time of a packet from start to finish. Ignoring processing time and using the terminology in the chapter, we have:

$$t_o = t_f + t_{ack} = \frac{8(512+60)}{10\times10^6} + \frac{64}{10\times10^6} = 4.64\times10^{-4}$$

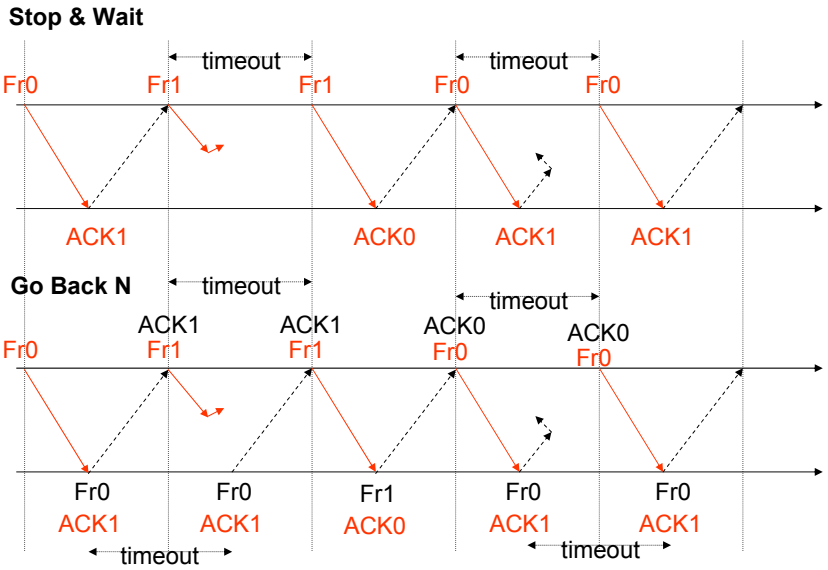
$$\eta_o = \frac{R_{eff}^0}{R} = \frac{\frac{n_f - n_o}{t_o}}{R} = \frac{\frac{8\times512}{4.64\times10^{-4}}}{10\times10^6} = 0.8828 = 88.3\%$$

One more source of overhead occurs because the last packet is not full. However, this additional overhead accounts for a very small fraction of the total overhead and does not affect the above result.

21. Compare the operation of Stop-and-Wait ARQ with bidirectional Go-Back-N ARQ with a window size of 1. Sketch out a sequence of frame exchanges using each of these protocols and observe how the protocols react to the loss of an information frame and to the loss of an acknowledgment frame.

Solution:

The figure below shows that the bidirectional Go-Back-N ARQ recovers from errors in the same time that Stop-and-Wait:



22. Consider the various combinations of communication channels with bit rates of 1 Mbps, 10 Mbps, 100 Mbps, and 1 Gbps over links that have roundtrip times of 10 msec, 1 msec, and 100 msec.

Solutions follow questions:

a. Find the delay-bandwidth product for each of the 12 combinations of speed and distance.

Delay-bandwidth (Megabits)

Bit Rate	Round Trip Time (msec)		
Mbps	100	10	1
1	0.1	0.01	0.001
10	1.0	0.10	0.010
100	10.0	1.00	0.100
1000	100.0	10.00	1.000

b. Suppose 32-bit sequence numbers are used to transmit blocks of 1000 bytes over the above channels. How long does it take for the sequence numbers to wrap around, that is, to go from 0 up to 2^m?

Block 1000 bytes
Sequence 32 bits 4294967296

Time for the sequence number to wrap around (sec)
Frame Time * 2^{32}=4294967296*8*1000/R seconds

Bit Rate	Round Trip Time (msec)		
Mbps	100	10	1
1	34359738.37	34359738.37	34359738.37
10	3435973.84	3435973.84	3435973.84
100	343597.38	343597.38	343597.38
1000	34359.74	34359.74	34359.74

c. Now suppose the 32-bit sequence numbers are used to count individual transmitted bytes. How long does it take for the sequence numbers to wrap around?

Time for the sequence number to wrap around (sec)
Byte Time * 2^{32}=4294967296*8/R seconds

Bit Rate	Round Trip Time (msec)		
Mbps	100	10	1
1	34359.74	34359.74	34359.74
10	3435.97	3435.97	3435.97
100	343.60	343.60	343.60
1000	34.36	34.36	34.36

The sequence number wraps around in much shorter time. At 1 Gbps the sequence number wraps around in only 34.36 seconds.

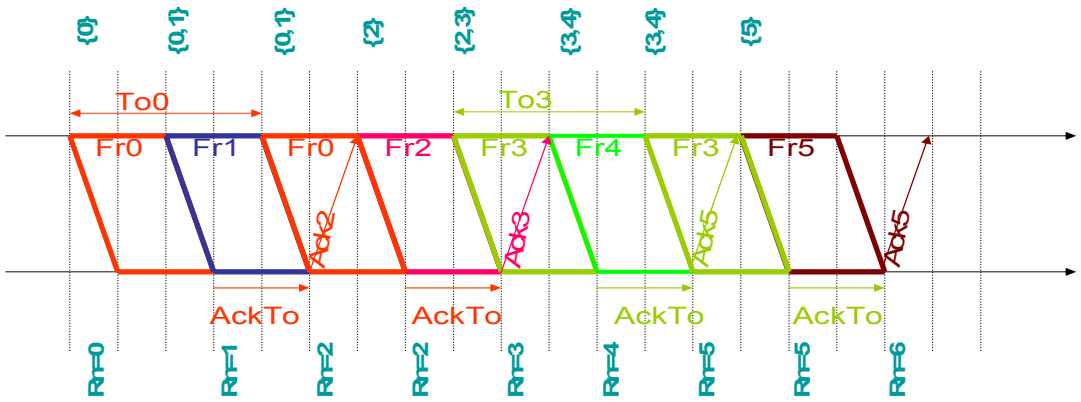
23. Consider a bidirectional link that uses Go-Back-N with $N = 7$. Suppose that all frames are one unit long and that they use a time-out value of 2. Assume the propagation is 0.5 unit and the processing time is negligible. Assume the ACK timer is one unit long. Assuming stations A and B begin with their sequence numbers set to zero, show the pattern of transmissions and associated state transitions for the following sequences of events:

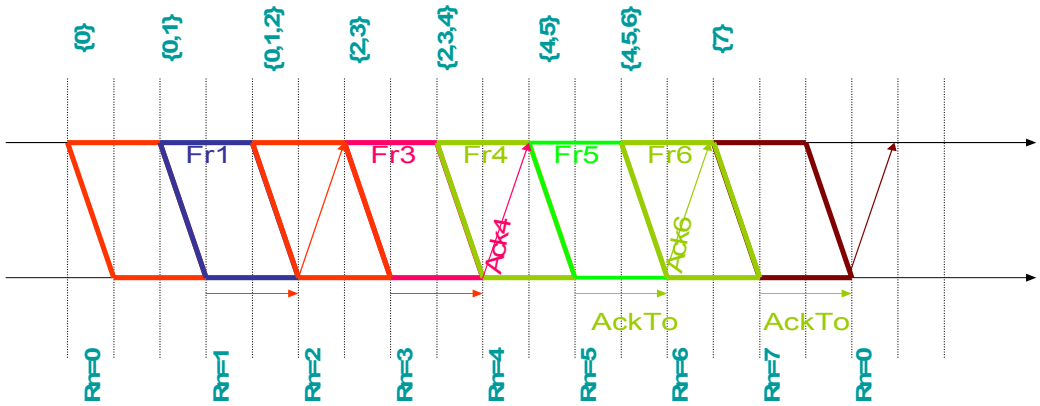
Solutions follow questions:

Go-Back-N with $N = 7$, $t_{timeout} = 2$, $t_{prop} = 0.5$, $t_{ACK} = 1$, $t_f = 1$. Each tick represents one half of a unit of time. The ACK timer delays the sending of an ACK to provide an opportunity for piggybacking.

a. Station A sends six frames in a row, starting at $t = 0$. All frames are received correctly.

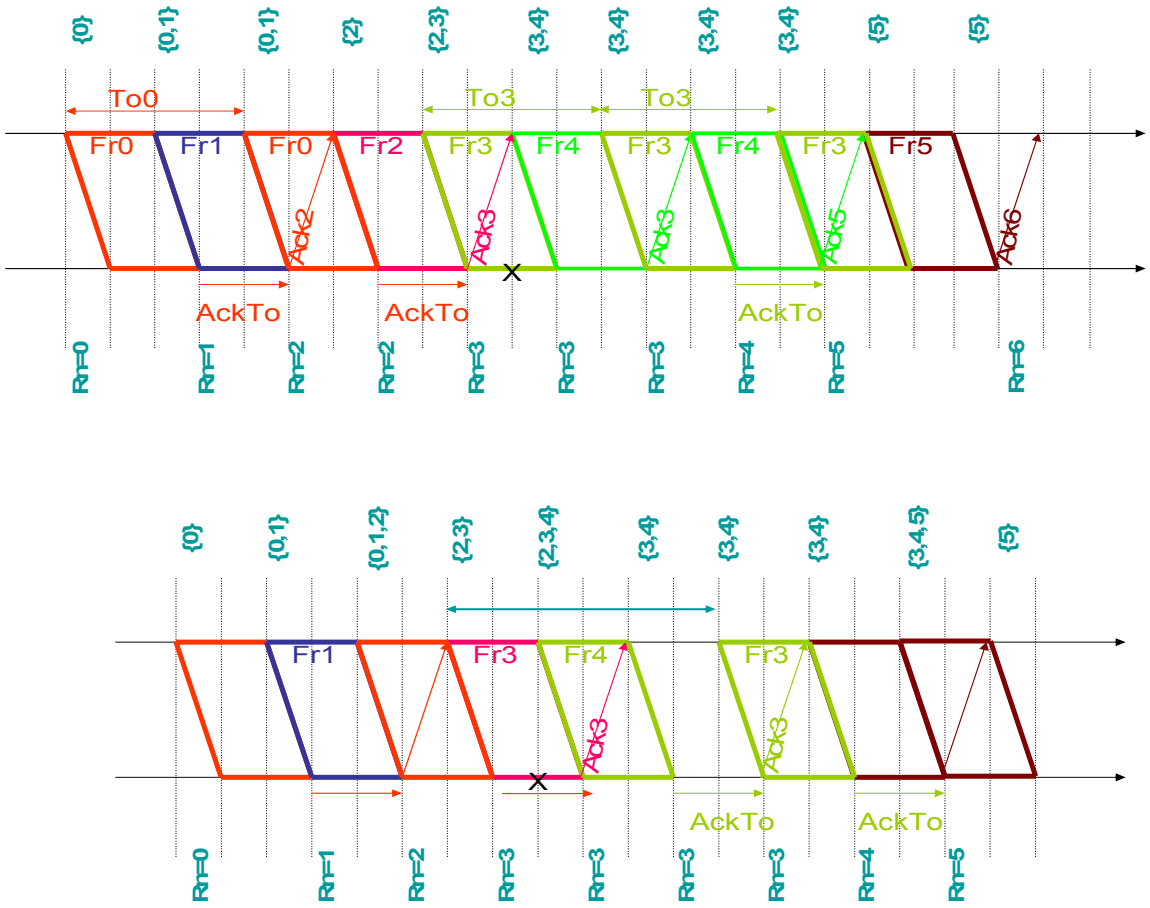
The first figure below shows that the timeout value is too short and cause needless retransmissions. The second figure below shows that by increasing the timeout value to 3 units, transmissions occur smoothly and efficiently.





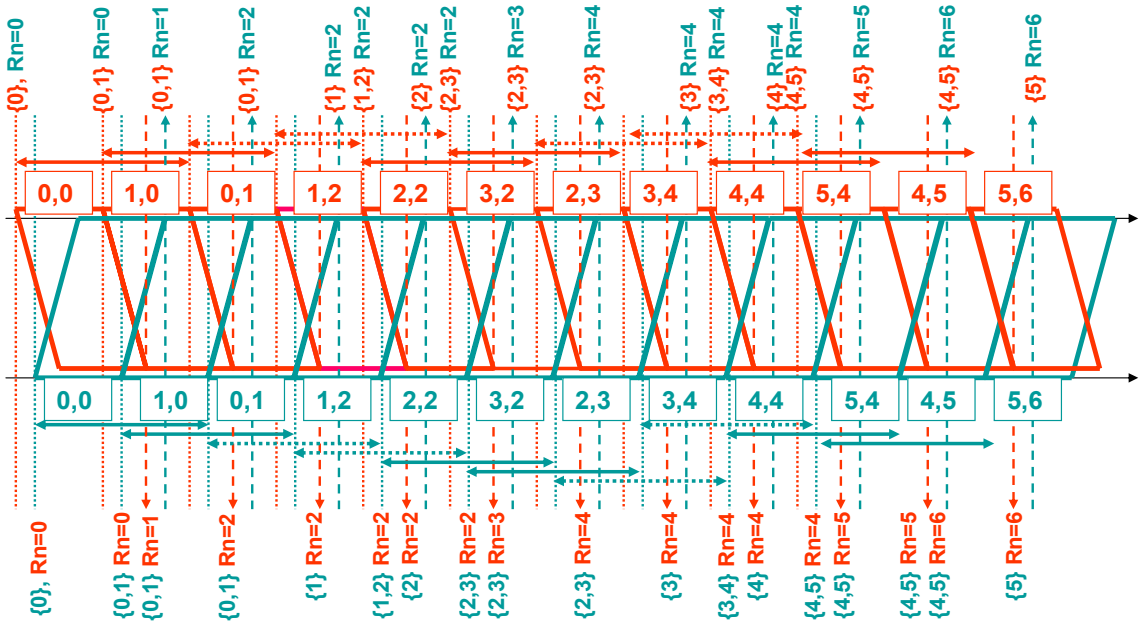
b. Station A sends six frame in a row, starting at $t = 0$. All frames are received correctly, except frame 3 is lost.

The following two figures show the sequence of frame exchanges with timeout values of 2 and 3 units. The system with the shorter timeout value initiates the recovery from error sooner, but cannot advance the window fast enough due to the short timeout value.



c. Station A sends six frames in a row, starting at $t = 0$. Station B sends six frames in a row starting at $t = 0.25$. All frames are received correctly.

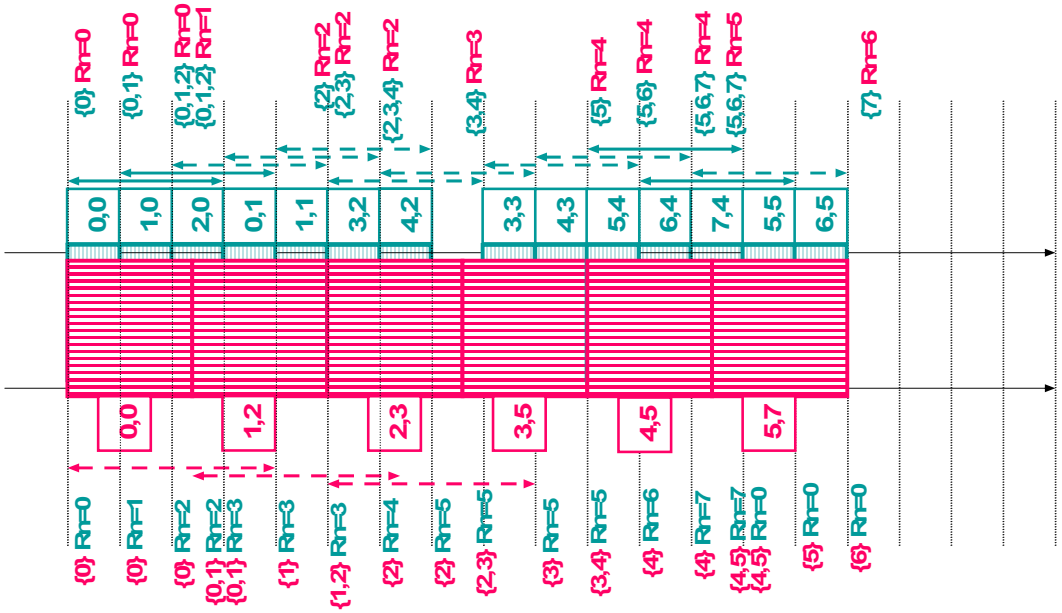
This problem shows that when piggybacking is used the timeout has too be somewhat longer. In the following figure we see that timeouts occur repeatedly causing needless retransmission of frames that have already arrived at the receiver. A timeout value of 4 is required to assure a smooth sliding forward of the transmitter's send window.

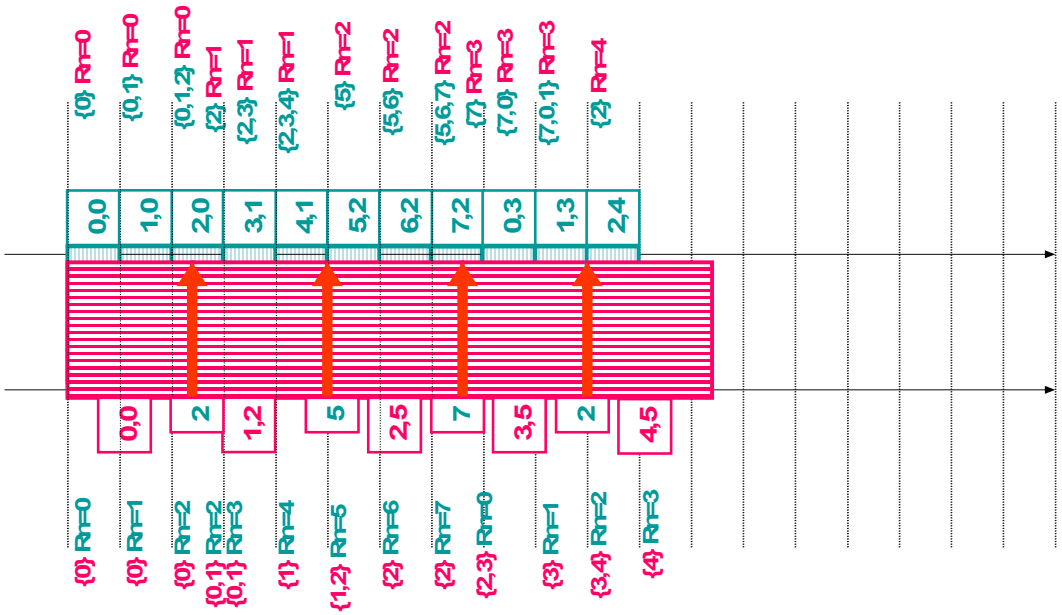


24. Consider a bidirectional link that uses Go-Back-N with $N = 3$. Suppose that frames from station A to station B are one unit long and that they use a time-out value of 2, and that frames in the opposite directions are 2.5 units long and that they use a time-out value of 4. Assume propagation and processing times are negligible, that the stations have an unlimited number of frames ready for transmission, and that all ACKs are piggybacked onto information frames. Assuming stations A and B begin with their sequence numbers set to zero, show the pattern of transmissions and associated state transitions that result if there are no transmission errors.

Solution:

The following shows the sequence of frames exchanged by stations A and B, as well as the set of unacknowledged frames, and the next expected frame at each station. Note that acknowledgments from station A to station B are frequent because of the short frame length and so station B sends frames in a continuous efficient manner. This is not the case for station A which receives acknowledgments after large delay and so frequently spends its time carrying out needless retransmissions. The solution to this problem is to insert ACK messages from station B to station A in between the long message transmissions. The second figure below shows that transmissions from A to B then flow smoothly.





25. Consider the Go-Back-N ARQ protocol.

Solutions follow questions:

a. What can go wrong if the ACK timer is not used?

When no traffic arrives at a receiver during bidirectional Go-Back-N ARQ, and the receiver has to send an ACK, it usually sends the ACK after the ACK timer expires. If the ACK timer is not used, there are only two options remaining:

1) The ACK must be sent immediately (that is, use piggybacking only if frame already available)

Although this will function correctly, it is an inefficient use of bandwidth in the general case.

2) The ACK must only be sent if it can be piggybacked

This is problematic if traffic arrives sporadically. The sender will wait a long time until a piggyback opportunity arises.

b. Show how the frame timers can be maintained as an ordered list where the time-out instant of each frame is stated relative to the time-out value of the previous frame.

Assume that the timer counts down from $t_{timeout}$. In order to have a separate timer for each frame, we need not implement N timers. Only the oldest frame can timeout. The system can save, for each frame, an arrival offset time that is related to the frame that preceded it and place these offsets in an ordered list based on the frame sequence numbers. If an ACK for the oldest frame arrives, the system simply increments the timer by the offset of the following frame in the list. If an ACK for any other frame arrives, the timer is incremented by the sum of all of the offsets in the list that are up to this newly acknowledged frame.

c. What changes if each frame is acknowledged individually instead of by using a cumulative acknowledgment (R_{next} acknowledges all frames up to $R_{next} - 1$)?

If each frame needs to be acknowledged individually, then the number of ACK messages will increase and the rate at which the transmission window can be increased will be reduced.

26. Suppose that instead of Go-Back-N ARQ, N simultaneous Stop-and-Wait ARQ processes are run in parallel over the same transmission channel. Each SDU is assigned to one of the N processes that is currently idle. The processes that have frames to send take turns transmitting in round-robin fashion. The frames carry the binary send sequence number as well as an ID identifying which ARQ process it belongs to. Acknowledgments for *all* ARQ processes are piggybacked onto *every* frame.

Solutions follow questions:

a. Qualitatively, compare the relative performance of this protocol with Go-Back-N ARQ and with Stop-and-Wait ARQ.

For simplicity assume that the time between consecutive frame transmissions in Stop-and-Wait corresponds to N consecutive transmissions without stopping. The parallel Stop-and-Wait procedure described above is an effective way to fill the transmission pipe without the additional complexity of Go-Back-N ARQ.

Vs. Go-Back N. Go-Back-N delivers frames in order. The parallel Stop-and-Wait protocol does not deliver frames in order, so additional processing is required if frames must be delivered in sequence. Because all the processes are independent, this protocol retransmits erroneous frames individually. In contrast, the Go-Back-N protocol retransmits a group of N frames. In this sense, the parallel protocol seems to perform similarly to a Selective Repeat process.

Vs. Stop-And-Wait. If Stop-and-Wait is used, the effective bit rate, without errors, will be N times less than the protocol described here. In fact, the larger N, the more efficient the protocol described is. At its worst case, where $N = 1$, it reduces to Stop-And-Wait.

b. How does the service offered by this protocol differ from that of Go-Back-N ARQ?

The parallel Stop-and-Wait protocol does not deliver frames in order, unless augmented by a frame resequencing scheme.

27. Write a program for the transmitter and the receiver implementing Go-Back-N ARQ over a data link that can introduce errors in transmission.

Solutions follow questions:

a. Identify what variables need to be maintained.

R_{next} : frame expected by receiver
S_{last} : oldest outstanding frame (back of window) at the transmitter
S_{recent} : most recently transmitted frame

b. The program loops continuously waiting for an event to occur that requires some action to take place. Identify the main events that can occur in the transmitter. Identify the main events that can occur in the receiver.

Go Back N Transmitter			
Event	**Condition**	**Action**	**Next State**
Request from upper layer	Not the last sequence number in nonempty send window	Prepare frame with S_{recent} and send, increment send sequence number; start timer	Ready
Request from upper layer	Last sequence number in nonempty send window	Prepare frame and send, increment send sequence number; start timer	Blocking
Arrival of error-free ACK frame	Correct sequence number, R_{next} is between S_{last} and S_{recent}	Slide window forward, $S_{last} = R_{next}$, max Sequence number = $S_{last} + W_S - 1$	Ready
Arrival of error-free ACK frame	Sequence number R_{next} is NOT between S_{last} and S_{recent}	Discard frame	Ready
Timeout Expires		Resend all frames from S_{last} onwards, reset timer	Ready
Arrival of erroneous frame		Discard frame	Ready
Request from upper layer		Requests are blocked	Blocking
Arrival of error-free ACK frame	Correct sequence number, R_{next} is between S_{last} and $S_{last} + W_S - 1$	Slide window forward, $S_{last} = R_{next}$, max Sequence number = $S_{last} + W_S - 1$	Ready
Arrival of error-free ACK frame	Sequence number R_{next} is NOT between S_{last} and S_{recent}	Discard frame	Blocking
Timeout Expires		Resend all frames from S_{last} onwards, reset timer	Blocking
Arrival of erroneous frame		Discard frame	Blocking

Go Back N Receiver				
Current State	**Event**	**Sequence Number**	**Action**	**Next State**
Ready	Arrival of error-free frame	Expected sequence number $S_{last} = R_{next}$	Accept frame, increment R_{next}, $R_{next} = R_{next} + 1$, and send ACK, deliver packet to higher layer	Ready
Ready	Arrival of error-free frame	Incorrect sequence number	Discard frame; send ACK with R_{next}	Ready
Ready	Arrival of erroneous frame		Discard frame	Ready

28. Modify the program in problem 5.27 to implement Selective Repeat ARQ.

Solution:

Selective Repeat Transmitter				
Current State	**Event**	**Sequence Number**	**Action**	**Next State**
Ready	Request from upper layer	Not the last sequence number in nonempty send window	Prepare frame with S_{recent} and send, increment send sequence number; start timer	Ready
Ready	Request from upper layer	Last sequence number in nonempty send window	Prepare frame and send, increment send sequence number; start timer	Blocking
Ready	Arrival of error-free ACK frame	Correct sequence number, R_{next} is between S_{last} and S_{recent}	Slide window forward, $S_{last} = R_{next}$, max sequence number = $S_{last} + W_S - 1$	Ready
Ready	Arrival of error-free NAK frame	Correct sequence number, R_{next} is between S_{last} and S_{recent}	Slide window forward, $S_{last}=R_{next}$, max Sequence number = $S_{last} + W_S - 1$, retransmit frame with R_{next}t	Ready
Ready	Arrival of error-free ACK/NAK frame	Sequence number R_{next} is NOT between S_{last} and S_{recent}	Discard frame	Ready
Ready	Timeout Expires		Resend frame corresponding to timer, reset associated timer	Ready
Ready	Arrival of erroneous frame		Discard frame	Ready
Blocking	Request from upper layer		Requests are blocked	Blocking
Blocking	Arrival of error-free ACK frame	Correct sequence number, R_{next} is between S_{last} and $S_{last} + W_S - 1$	Slide window forward, $S_{last}=R_{next}$, max sequence number = $S_{last} + W_S - 1$	Ready
Blocking	Arrival of error-free NAK frame	Correct sequence number, R_{next} is between S_{last} and S_{recent}	Slide window forward, $S_{last}=R_{next}$, max sequence number = $S_{last} + W_S - 1$, retransmit frame with R_{next}	Ready
Blocking	Arrival of error-free NAK frame	Correct sequence number, $R_{next} = S_{last}$	retransmit frame with R_{next}	Blocking
Blocking	Arrival of error-free ACK/NAK frame	Sequence number R_{next} is NOT between S_{last} and S_{recent}	Discard frame	Blocking
Blocking	Timeout expires		Resend frame corresponding to timer, reset associated timer	Blocking
Blocking	Arrival of erroneous frame		Discard frame	Blocking

Selective Repeat Receiver				
Current State	**Event**	**Sequence Number**	**Action**	**Next State**
Ready	Arrival of error-free frame	Sequence number is R_{next}	Accept frame; slide window forward $R_{next} = R_{next} + k$, and send ACK, deliver k packets to higher layer	Ready
Ready	Arrival of error-free frame	Sequence number is in the receive window, that is, between $R_{next}+1$ and $R_{next} + W_R - 1$	Accept and buffer frame; and send ACK	Ready
Ready	Arrival of error-free frame	Sequence number is outside the receive window,	Discard frame, send ACK with R_{next}	Ready
Ready	Arrival of erroneous frame		Discard frame	Ready

29. Three possible strategies for sending ACK frames in a Go-Back-N setting are as follows: send an ACK frame immediately after each frame is received, send an ACK frame after every other frame is received, and send an ACK frame when the next piggyback opportunity arises. Which of these strategies are appropriate for the following situations?

Solutions follow questions:

a. An interactive application produces a packet to send each keystroke from the client; the server echoes each keystroke that it receives from the client.

Since each keystroke is echoed, there will always be a piggyback opportunity. Thus, the piggyback method should be used. Indeed, the echo packet constitutes an acknowledgment.

b. A bulk data transfer application where a server sends a large file that is segmented in a number of full-size packets that are to be transferred to the client.

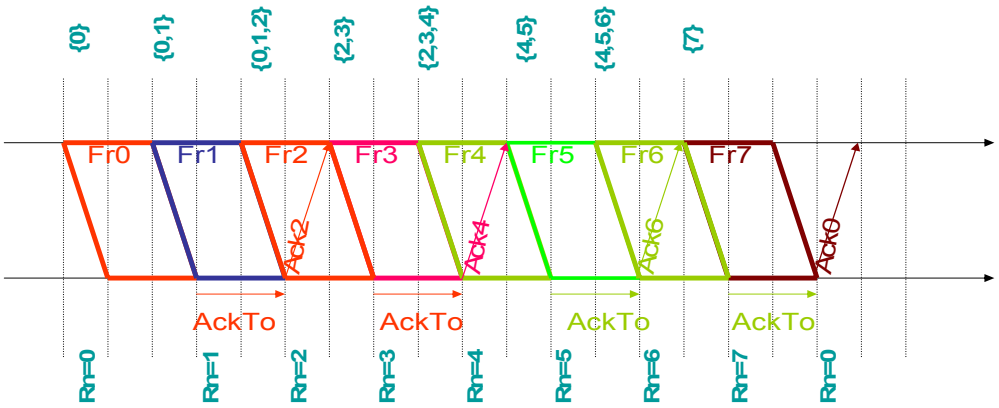
The upstream traffic *to* a server is generally much less than the downstream traffic. Thus, the piggybacking method is non-ideal in this case. If the channel has a low probability of error, the alternating ACK method is better, as it saves bandwidth. However, if the connection causes frequent errors, every frame should be acknowledged. Additional overhead traffic is caused by the ACK frames, but will compensated by the bandwidth savings that will arise when the errors are discovered more quickly.

30. Consider a bidirectional link that uses Selective Repeat ARQ with a window size of $N = 4$. Suppose that all frames are one unit long and use a time-out value of 2. Assume that the one-way propagation delay is 0.5 time unit, the processing times are negligible, and the ACK timer is one unit long. Assuming station A and B begin with their sequence numbers set to zero, show the pattern of transmissions and associated state transitions for the following sequences of events:

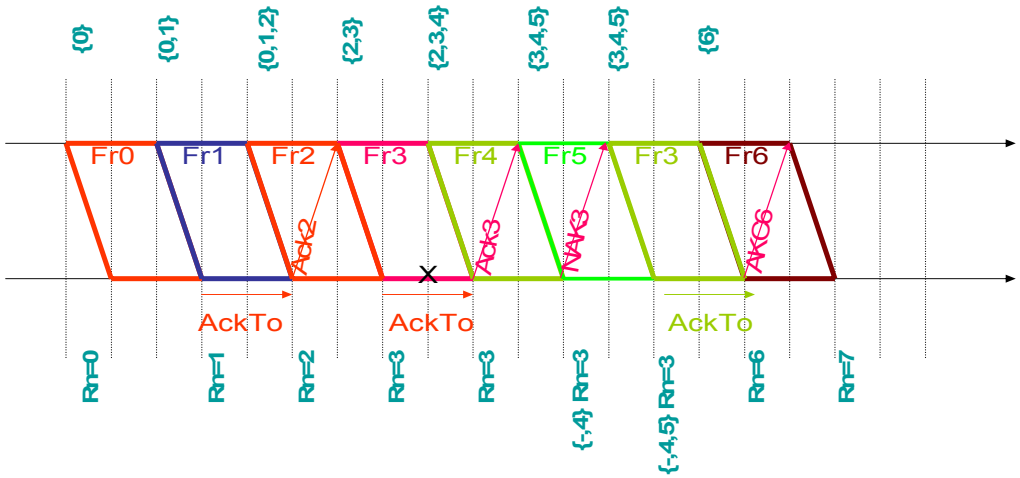
Solutions follow questions:

a. Station A sends six frames in a row, starting at $t = 0$. All frames are received correctly.

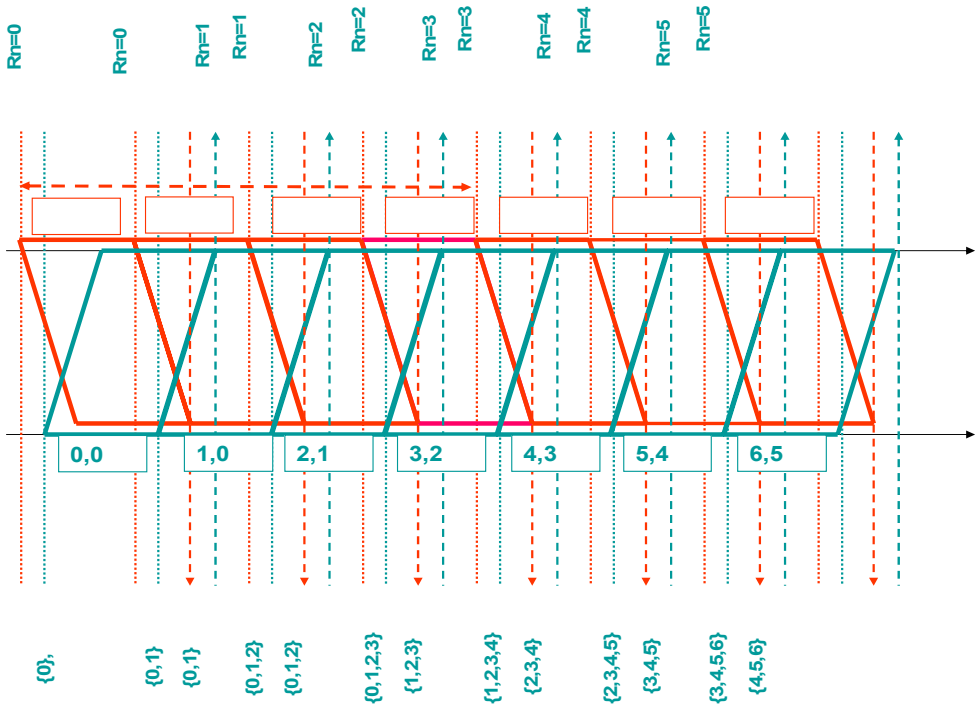
A timeout value of 2 causes the transmitter to resend frames before the window size of 4 is exhausted. We will assume a timeout value of 4 in the following solutions.



b. Station A sends six frame in a row, starting at $t = 0$. All frames are received correctly, except frame 3 is lost.



c. Station A sends six frames in a row, starting at $t = 0$. Station B sends six frames in a row starting at $t = 0.25$. All frames are received correctly.



31. In the chapter we showed that if the transmit and receive maximum window sizes are both equal to half the available sequence number space, then Selective Repeat ARQ will work correctly. Rework the arguments presented in the chapter to show that if the sum of the transmit and receive maximum window sizes equals the available sequence number space, then Selective Repeat ARQ will work correctly.

Solution:

Assume $W_s + W_R = 2^m$ and assume that the current send window is 0 to $W_S - 1$. Suppose also that the receive window is 0 to $W_R - 1$. Now suppose that frame 0 is received correctly but that the

acknowledgment for frame 0 is lost. The transmitter can transmit new frames only up to frame $W_S - 1$. Depending on which transmissions arrive without error, R_{next} will be in the range between 1 and W_s while $R_{next} + W_R - 1$ will be in the range of 1 to $W_R + W_S - 1$. The maximum value of R_{next} occurs when frames 0 through $W_S - 1$ are received correctly, in which case the value of R_{next} is W_S and the value of $R_{next} + W_R - 1$ increases to $W_R + W_S - 1$. Crucially, the receiver will not receive frame $W_R + W_S$ until the acknowledgment for frame 0 has been received at the transmitter. Any receipt of frame 0 prior to frame $W_R + W_S$ indicates a duplicate transmission of frame 0. Therefore, the sum of the maximum window sizes is 2^m.

32. Suppose that Selective Repeat ARQ is modified so that ACK messages contain a list of the next *m* frames that it expects to receive.

Solutions follow questions:

a. How does the protocol need to be modified to accommodate this change?

First, the frame header needs to be modified to accommodate the list of frames to receive. It can be a fixed or a variable number of slots. NAK won't be necessary because the receiver explicitly indicates which frames need to be transmitted. Second, the transmitter operation must change to retransmit frames according to the received list. If the list contains the m oldest frames that are yet to be received, then the list can be used to skip retransmissions of frames that have already been received.

b. What is the effect of the change on protocol performance?

The performance will increase in cases with high error rate or in cases where the delay is high. A single frame can ask for the retransmission of several frames. The drawback is the overhead in the header and the increased protocol complexity relative to pure Selective-Repeat ARQ.

33. A telephone modem is used to connect a personal computer to a host computer. The speed of the modem is 56 kbps and the one-way propagation delay is 100 ms.

Solutions follow questions:

a. Find the efficiency for Stop-and-Wait ARQ if the frame size is 256 bytes; 512 bytes. Assume a bit error rate of 10^{-4}.

First we have the following:

$P_f = 1 - (1 - 10^{-4})^{n_f}$
$n_f = 256 \times 8 = 2048$ or $n_f = 512 \times 8 = 4096$
$t_{prop} = 100$ ms

$n_o = 0$
$n_a = 64$ bits
$t_{proc} = 0$

Using the results in Equation 5.4,

$$\eta = (1 - P_f)\frac{1 - \frac{n_o}{n_f}}{1 + \frac{n_a}{n_f} + \frac{2(t_{prop} + t_{proc})}{n_f}R}$$

= 0.125 ($n_f = 2048$)
= 0.177 ($n_f = 4096$)

b. Find the efficiency of Go-Back-N if three-bit sequence numbering is used with frame sizes of 256 bytes; 512 bytes. Assume a bit error rate of 10^{-4}.

Given that $W_S = 2^3 - 1 = 7$, we can calculate that the window size is:

$$\frac{n_f \times W_S}{R} = 256ms$$

Since this is greater than the round trip propagation delay, we can calculate the efficiency by using the results in Equation 5.8.

$$\eta = (1 - P_f)\frac{1 - \frac{n_o}{n_f}}{1 + (W_S - 1)P_f}$$

= 0.385 (n_f = 2048)

= 0.220 (n_f = 4096)

34. A communications link provides 1 Mbps for communications between the earth and the moon. The link is used to send color images from the moon. Each image consists of 10,000 x 10,000 pixels, and 16 bits are used for each of the three-color components of each pixel.

Solutions follow questions:

a. How many images per second can be transmitted over the link?

The number of images that can be transmitted per second is:

$$1 \times 10^6 \frac{bits}{\sec} \Big/ (10000^2 \times 16 \times 3 \frac{bits}{image}) = 2.1 \times 10^{-4} images / \sec ond$$

b. If each image is transmitted as a single block, how long does it take to get an acknowledgment back from earth? The distance between earth and the moon is approximately 375,000 km.

The total time to get an acknowledgment from earth, assuming that $t_{ACK} << t_f$ is:

$$t_O = t_f + 2t_{prop} = \frac{(10000^2 \times 16 \times 3 \frac{bits}{image})}{1 \times 10^6 \frac{bits}{\sec}} + 2\frac{375000 \times 10^3 m}{3 \times 10^8 \frac{m}{\sec}}$$

$$= 4800 + 2.50 = 4802.5 \sec / image$$

Note that if each image is transmitted in a single block, t_{prop} becomes insignificant compared to t_f.

c. Suppose that the bit error rate is 10^{-5}, compare Go-Back-N and Selective Repeat ARQ in terms of their ability to provide reliable transfer of these images from the moon to earth. Optimize your frame size for each ARQ protocol.

The difficulty here is that if we transmit an entire image as a single block, then the average number of errors in a block is 48 x 10^8 x 10^{-5} = 48000. Thus every transmission fails. Clearly we need to transmit using a smaller frame size, say n. Qualitatively, the following happens as we vary n. If n is very small, then the probability of frame error is small, but the overhead due to the header n_o will become significant. If n becomes too large, then the efficiency drops because of frequent retransmissions. Thus there must be an intermediate value of n that optimizes efficiency. We will find this value by trial and error. We will assume a header overhead of 64 bits in a frame of size n greater than 64 bits.

Each frame carries n bits of information, so the required window size for a propagation delay of 2.5 seconds is then W_s=2.5x10^6/n+1 The probability of frame error is then

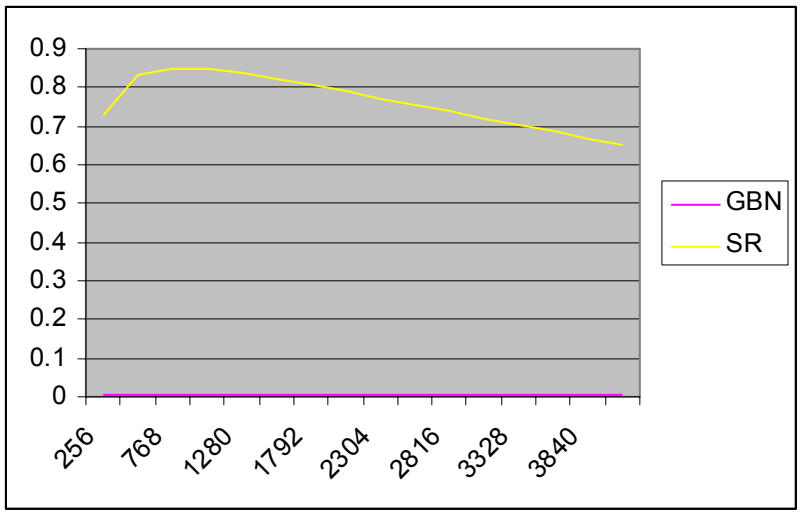
$$P_f = 1-(1-10^{-5})^n \approx 1-e^{-n10^{-5}}$$

The efficiency for Go-Back-N is given by:

$$\eta_{GBN} = \frac{(1-P_f)(1-\frac{64}{n})}{1+(W_s-1)P_f} = \frac{(1-\frac{64}{n})e^{-n10^{-5}}}{1+(\frac{2.5x10^6}{n})(1-e^{n10^{-5}})}$$

Using similar assumptions, the efficiency of Selective Repeat ARQ is given by

$$\eta_{SR} = (1-P_f)(1-\frac{64}{n}) = (1-\frac{64}{n})e^{-n10^{-5}}$$



The figure above shows the efficiency of Go-Back-N and Selective Repeat ARQ as a function of frame size. Go-Back-N has very low efficiency (always below 1%) for all values of n. Selective Repeat achieves a maximum of about 85% efficiency at around *n*=768 bits.

Note that the optimum value of *n* is dependent on the value of n_o.

35. Two computers are connected by an intercontinental link with a one-way propagation delay of 100 ms. The computers exchange 1-Megabyte files that they need delivered in 250 ms or less. The transmission lines have a speed of R Mbps and the bit error rate is 10^{-8}. Design a transmission system by selecting the bit rate R, the ARQ protocol, and the frame size.

Solution:

Using Selective Repeat ARQ for efficiency's sake and assuming that the overhead n_o = 64 bits, we need to select n_f and R to ensure delivery in 250 ms or less.

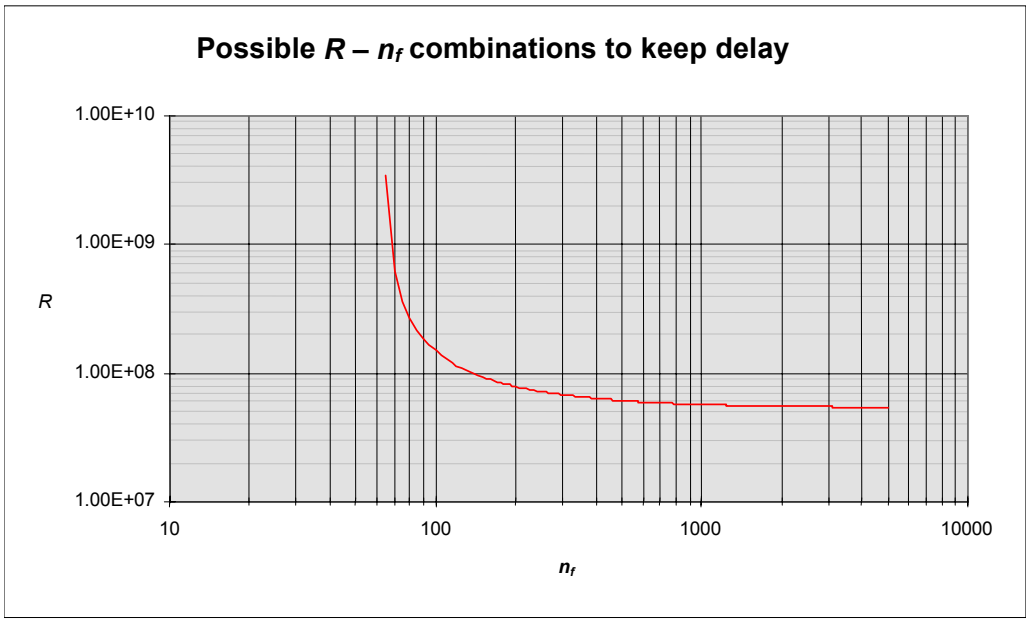
$$\text{Total delay} = 2t_{prop} + \frac{8\times10^6}{R_{eff}^o} = 200\times10^{-3} + \frac{8\times10^6}{\eta R} < 250\times10^{-3}$$

$$\eta R > 6.25\times10^{-9}$$

From Equation 5.11 we see that

$$(1-10^{-8})^{n_f}(1-\frac{64}{n_f})R > 53.33\times10^6$$

The allowable region is shown below as the area under the graph.



36. Find the optimum frame length n_f that maximizes transmission efficiency for a channel with random bit errors by taking the derivative and setting it to zero for the following protocols:

Solutions follow questions:

a. Stop-and-Wait ARQ.

$$\eta = (1-P_b)^{n_f}\frac{n_f - n_o}{n_f + n_a + 2(t_{proc} + t_{prop})R}$$

Let $a = (1 - P_b)$
Let $b = 2\,(t_{proc} + t_{prop})\,R + n_a$

$$\frac{d}{dn_f}[a^{n_f}\,\frac{n_f - n_o}{n_f + b}] = \frac{a^{n_f}[\ln a(n_f - n_o) + 1](n_f + b) - a^{n_f}(n_f - n_o)}{(n_f + b)^2} = 0$$

$$a^{n_f}\{\ln a[n_f^2 + (b - n_o)n_f - n_o b] + n_f + b - n_f + n_o\} = 0$$

$$[n_f^2 + (b - n_o)n_f - n_o b] + \frac{b + n_o}{\ln a} = 0$$

$$n_f = \frac{(n_o - b) \pm \sqrt{(b - n_o)^2 - 4\left[\frac{b + n_o}{\ln a} - b n_o\right]}}{2}$$

b. Go-Back-N ARQ.

$$\eta = (1 - P_b)^{n_f}\,\frac{n_f - n_o}{n_f + n_f[1 - (1 - P_b)^{n_f}][\frac{2R(t_{proc} + t_{prop})}{n_f}]}$$

Let $a = (1 - P_b)$
Let $b = 2R\,(t_{proc} + t_{prop})$ and use the approximation $1 - (1 - p)^n \approx np$, (which is valid when np <<1), then

$$\eta = \frac{a^{n_f}(n_f - n_o)}{n_f + n_f P_b b} = \frac{a^{n_f}(n_f - n_o)}{n_f[1 + P_b b]}$$

Taking the derivative and equating the numerator to zero, we find that:

$$0 = \frac{d\eta}{dn_f} = \frac{d}{dn_f}\left[\frac{a^{n_f}(n_f - n_o)}{n_f}\right] = \frac{(a^{n_f}\ln a(n_f - n_0) + a^{n_f})n_f - a^{n_f}(n_f - n_0)}{{n_f}^2}$$

which leads to the quadratic equation

$$n_f^2 - n_o n_f + n_o / \ln a = 0$$

which gives the solution:

$$n_f = \frac{n_o \pm \sqrt{{n_o}^2 - 4\frac{n_o}{\ln a}}}{2}$$

c. Selective Repeat ARQ.

Borrowing the result from problem 5.34c:

$$\eta = (1 - P_b)^{n_f}\,(1 - \frac{n_0}{n_f})$$

Except for a scale factor, this equation is the same as the approximation for η found in part (b). The solution for the optimum frame size has the same form.

d. Find the optimum length for a 1 Mbps channel with 10 ms reaction time, 25-byte overhead, 25-byte ACK frame, and $p = 10^{-4}$, 10^{-5}, and 10^{-6}.

$R = 1\text{x}10^{6}$,

$n_o = 8 \text{ x } 25 = 200$,

$n_a = 8 \text{ x } 25 = 200$,

$P_b = 10^{-4}, 10^{-5}, 10^{-6}$

$a = 1 - P_b$, $\ln a = -1\text{x}10^{-3}, -1\text{x}10^{-4}, -1\text{x}10^{-5}$

$t_{prop} + t_{proc} = 10$ ms, $b = 2\,(t_{proc} + t_{prop})\,R + n_a = 2 \text{ x } 10^{-2} \text{ x } 1\text{x}10^{6} + 200 = 20200$

P	10^{-4}	10^{-5}	10^{-6}
Stop and Wait ARQ	1156	7552	36303
Go back N ARQ	558	1517	4573
Selective Repeat	558	1517	4573

37. Suppose station A sends information to station B on a data link that operates at a speed of 10 Mbps, and that that station B has a 1-Megabit buffer to receive information from A. Suppose that the application at station B reads information from the receive buffer at a rate of 1 Mbps. Assuming that station A has an unlimited amount of information to send, sketch the sequence of transfers on the data link if Stop-and-Wait ARQ is used to prevent buffer overflow at station B. Consider the following cases:

a. One-way propagation delay is 1 microsecond.

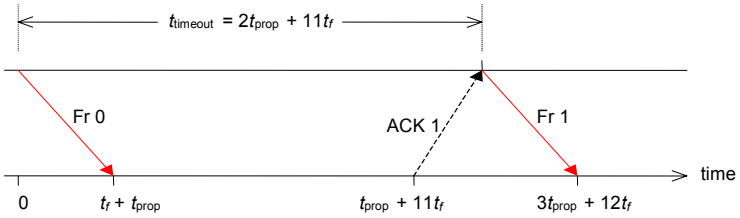
b. One-way propagation delay is 1 ms.

c. One-way propagation delay is 100 ms.

Note: The solution to this problem is currently being reviewed and may be revised.

Solution:

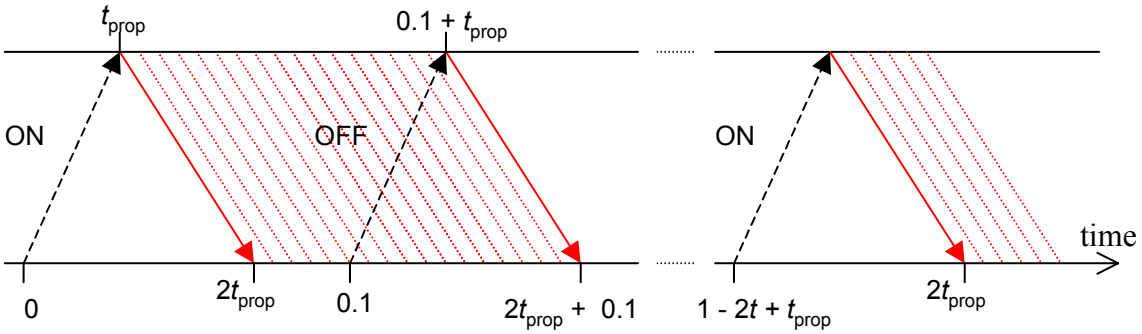
The issue in this problem is that the receiver clears its buffer at 1/10 the rate that the channel can put information into the buffer. Consequently, acknowledgments must be withheld from the transmitter to allow the receiver time to clear the buffer. For the sake of simplicity assume a frame is 1 Mbit long, then it takes 1 Mbit/10 Mbps = 100 ms to transmit the frame, which arrives in its entirety at the receiver at time 100 ms + t_{prop} as shown in the figure below.



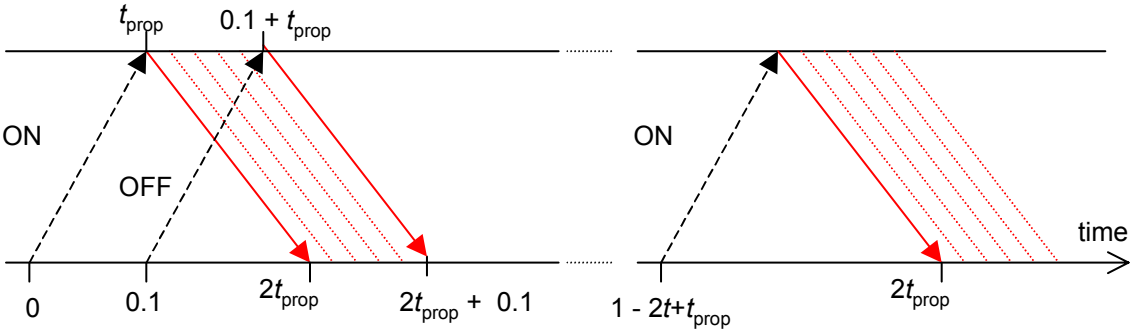
38. Redo problem 5.37 using Xon/Xoff flow control.

Solution:

If t_{prop} is less than 50ms (parts (a) and (b)), the following picture holds.



If t_{prop} is greater than 50ms, the following picture is more indicative of the sequence of events:



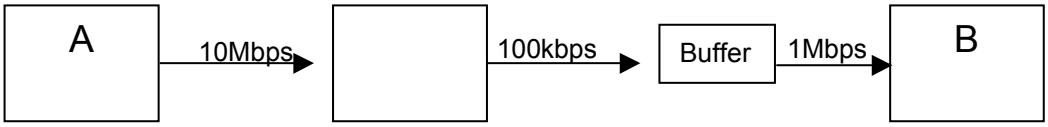
39. Suppose station A sends information to station B over a two-hop path. The data link in the first hop operates at a speed of 10 Mbps, and the data link in the second hop operates at a speed of 100 kbps. Station B has a 1-Megabit buffer to receive information from A, and the application at station B reads information from the receive buffer at a rate of 1 Mbps. Assuming that station A has an unlimited amount of information to send, sketch the sequence of transfers on the data link if Stop-and-Wait ARQ is used on an end-to-end basis to prevent buffer overflow at station B.

Note: The solution to this problem is currently being reviewed and may be revised.

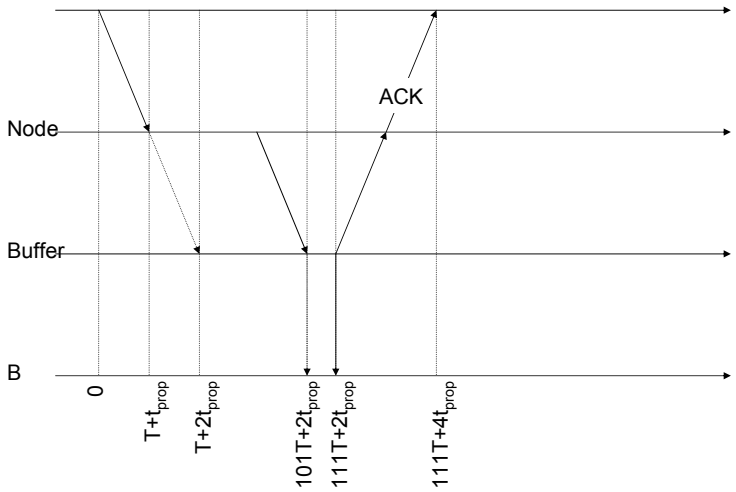
a. One-way propagation delay in data link 1 and in data link 2 are 1 ms.

b. One-way propagation delay in data link 1 and in data link 2 are 100 ms.

Solution:



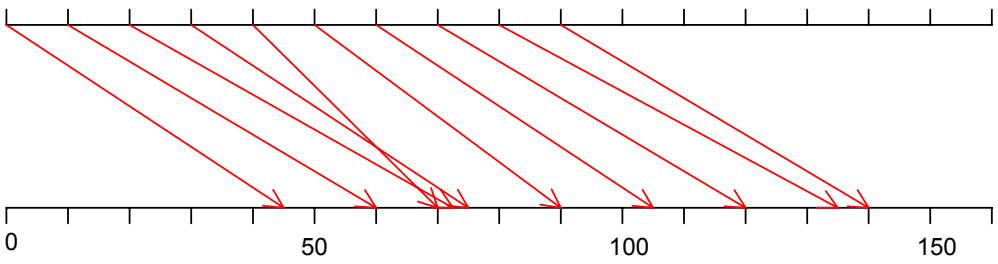
Assume that a frame is n_f bits long, then the frame transmission time in link 1 is $t_1 = n_f/10^7$ seconds, and $t_2 = n_f/10^5$ seconds on the second link as shown in the figure below. Therefore the time to transmit a frame over the second link is 100 times greater than the time to transmit over the first link. For example if n_f = 10000 bits, then T=t_1 = 1 ms and t_2 =100 ms=100T, and similarly if n_f = 1000000 bits, then t_1 = 100 ms and t_2 =10 seconds. Consequently, the node between link 1 and link 2 must be capable of buffering the bits that arrive in link 1 but which must wait for transmission over link 2. If we assume that the frame is buffered until it has completely arrived, then it will take $t_3 = n_f/10^6$ =10T seconds to read out. In the above examples, we have t_3 = 10 ms and 1 second respectively. The ACK message for Stop-and-Wait can then be sent. Let t_f=t_1+t_2+t_3=111T. The sequence of events is then as follows: a frame that begins transmission at time 0 will be received in its entirety at node B at time t_f +2 t_{prop}; an ACK message is send thereafter, and the message arrives at the transmitter at approximately time t_f+4t_{prop}.



40. A sequence of fixed-length packets carrying digital audio signal is transmitted over a packet network. A packet is produced every 10 ms. The transfer delays incurred by the first 10 packets are 45 ms, 50 ms, 53 ms, 46 ms, 30 ms, 40 ms, 46 ms, 49 ms, 55 ms, 51 ms.

Solutions follow questions:

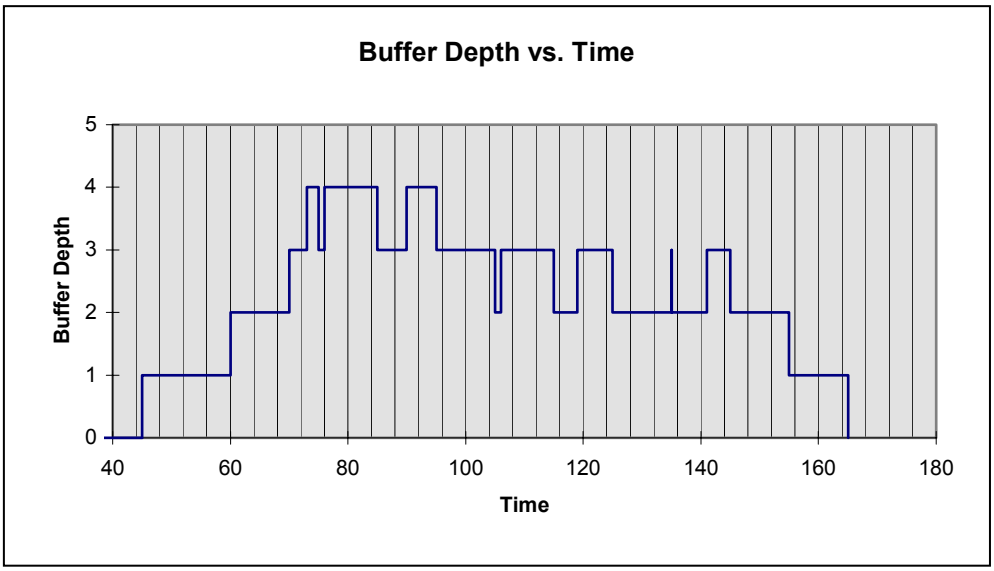
a. Sketch the sequence of packet transmission times and packet arrival times.



b. Find the delay that is inserted at the receiver to produce a fixed end-to-end delay of 75 ms.

Packet #	0	1	2	3	4	5	6	7	8	9
Transmission Time	0	10	20	30	40	50	60	70	80	90
Delay	45	50	53	46	30	40	46	49	55	51
Arrival Time	45	60	73	76	70	90	106	119	135	141
Desired playout time	75	85	95	105	115	125	135	145	155	165
Delay Inserted	**30**	**25**	**22**	**29**	**45**	**35**	**29**	**26**	**20**	**24**

c. Sketch the contents of the buffer at the receiver as a function of time.



41. Consider an application in which information that is generated at a constant rate is transferred over a packet network so timing recovery is required at the receiver.

Solutions follow questions:

a. What is the relationship between the maximum acceptable delay and the playout buffer?

There are several components in the delay experienced by the information: (1) packetization delay, the time the first byte in a packet payload waits until the packet is full; (2) network transfer delay, the time to traverse the network; and (3) playout delay, the additional delay inserted at the receiver. The sum of these three components should not exceed the maximum acceptable delay. In particular, the length of the playout buffer cannot be longer than the product of the maximum delay and the bit rate of the information.

b. What is the impact of the bit rate of the application information stream on the buffer requirements?

The maximum delay incurred by the buffering is explicitly equal to

Max buffer delay = buffer size (in bits) / playout rate

Since the playout rate is equal to the bit rate of the application information stream, we see that for a given acceptable maximum playout delay, the buffer size is directly determined by the playout rate.

c. What is the effect of jitter on the buffer size design?

In general, to reduce playback delay, the buffer length should be kept as small as possible. However, the buffer must be able to accommodate the jitter of the incoming traffic stream.

Jitter has the effect of making the incoming data rate variable. If the stream has high jitter (i.e. high delay variations), then the buffer must be made large to accommodate times when the incoming data rate may be higher than the playback rate. In addition the playout delay should ensure that enough packets are buffered so at no time will the buffer become empty and cause "starvation" of the playout system.

42. A speech signal is sampled at a rate of 8000 samples/second. Packets of speech are formed by packing 10 ms worth of samples into each payload. Timestamps are attached to the packets prior to transmission and used to perform error recovery at the receiver. Suppose that the time stamp is obtained by sampling a clock that advances every Δ seconds. Is there a minimum value that is required for Δ? If so what is it?

Solution:

The clock is sampled every 10 ms to produce a timestamp. Clearly, we require that the clock advance at least once between samples, that is, $\Delta < 10$ ms, in order to ensure a unique timestamp for each packet. In addition if timestamps consist of m bits, we do not want the clock to advance so quickly that the timestamp counter wraps around in one sample time, that is, $2^m \times \Delta >> 10$ ms.

The timestamps can be used to resequence packets at the receiver. However if the timestamps are to be used for error detection, a Δ of just under 10 ms is not sufficient. To illustrate, suppose that the clock advances every $\Delta = 9$ ms. Then the times at which the clock advances will be as follows:

Clock tick:	0	9	18	27	36	45	54	63	72	81	90
Count:	0	1	2	3	4	5	6	7	8	9	10

The time stamps that result from sampling every 10 ms will be:

Sample:	0	10	20	30	40	50	60	70	80	90	100
Stamp:	0	1	2	3	4	5	6	7	8	10	11

We see that the sample at time 90 will have a time stamp of 10, which is 2 units larger than the time stamp at time 80. If a packet is lost, a gap in the timestamps will be detected by the receiver. However for the example, the receiver may not be able to decide whether a packet was lost or whether the phenomenon above occurred.

While it is true that this problem can be overcome because of the periodic nature of the above phenomenon, a more reliable solution is to place the constraint that $\Delta < 5$ms. With this constraint in place each timestamp will be offset by 2Δ or possibly 3Δ (in the case of an extra skipped clock pulse). If a packet is lost, a timestamp offset of 4Δ or possibly 5Δ will be detected at the receiver, which is completely unambiguous.

43. Suppose that UDP is used to carry the speech signal in the previous problem. What is the total bit rate consumed by the sequence of UDP PDUs? What is the percent overhead?

Solution:

Number of data bits in each PDU = $64 \times 10^3 \times 10 \times 10^{-3} = 640$ bits

UDP header size = 8 x 8 bytes = 64 bits

Total bit rate = (640 + 64) / ($10x10^{-3}$) = 704 x 10^2 bps = 70.4 Kbps

Overhead = [64 / (640 + 64)] x100= 9.1%

44. Suppose that PDUs contain both timestamps and sequence numbers. Can the timestamps and sequence numbers be combined to provide a larger sequence number space? If so, should the timestamps or the sequence numbers occupy the most significant bit locations?

Solution:

The combination of timestamps and sequence numbers help protect a receiver against sequence number wraparound in very high speed networks. If the transmission speed is sufficiently high, it is possible for a transmitter to re-use a sequence number in a relatively short period of time. The timestamp can be used to distinguish between frames (segments) with the same sequence number. Here is how: the sequence numbers can be viewed as being obtained by looking at the time in a wall clock; the timestamp can correspond to the date. The combination of date and clock time allow the receiver to distinguish between a new segment (with a given sequence number) and an old retransmitted segment (with the same sequence number). In order for this to work, the date should advance at least once in the time it takes for the sequence to wrap around, i.e. the date should advance every 24 hours. This combination of timestamps and sequence numbers is called the PAWS algorithm for "protection against wrapped sequence numbers."

45. Consider the timestamp method for timing recovery discussed in the chapter.

Solutions follow questions:

a. Find an expression that related the difference frequency Δf to the number of cycles M and N.

$$\Delta f = f_n - f_s = f_n(1 - \frac{M}{N})$$

b. Explain why only M needs to be sent.

f_n is globally known and N is agreed on beforehand (as a standard or else during connection setup). Thus, only M needs to be sent.

c. Explain how the receiver uses this value of M to control the playout procedure.

The procedure plays out frames at a rate:

$$f_r = f_n - \Delta f = f_n - fn\ (1 - \frac{M}{N}) = \frac{M}{N} f_n$$

46. In Figure 5.32, determine the number of bytes exchanged between client and server in each direction.

Solution:

12 bytes from server to client in frame 4, and 3 bytes from client to server in frame 5. All other exchanges involve only control information in the header.

47. Suppose that the delays experienced by segments traversing the network is equally likely to be any value in the interval [50 ms, 75 ms].

Solutions follow questions:

a. Find the mean and standard deviation of the delay.

The delay lies between interval [50ms, 75ms] and is a uniform random variable. The mean is:

$E[X] = (50 + 75) / 2 = 62.5$ ms.

The standard deviation of delay is:

$STD[x] = VAR[x]^{1/2} = [(75 - 50)^2 / 12]^{1/2} = 7.217$

b. Most computer languages have a function for generating uniformly distributed random variables. Use this function in a short program to generate random times in the above interval. Also, calculate t_{RTT} and d_{RTT} and compare to part (a).

See below for sample program written in C.

```
/* Communication Networks - Chapter 5    */
/* Question 47 (b)                       */
/* Description - Generate a random value */
/* between 50 to 75 ms.  Calculate t_RTT */
/* and d_RTT                             */
/* The min, max and avg value of t_RTT   */
/* and d_RTT are also recorded           */

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int main (void)
{
  int i;
  float temp, t_n, t_rtt_new, t_rtt_old;
  float d_rtt_new, d_rtt_old;
  float t_rtt_min, t_rtt_sum, t_rtt_max;
  float d_rtt_min, d_rtt_sum, d_rtt_max;

  const float alpha = 0.875;
  const float beta = 0.25;

  srand (time(NULL));
  t_rtt_old = 0;
  d_rtt_old = 0;

  t_rtt_sum = t_rtt_max = 0;
  d_rtt_sum = d_rtt_max = 0;

  t_rtt_min = d_rtt_min = 500;

  for (i = 0; i < 500; i++)
  {

    /* Generate a random value between 0 to 1 */
    temp = (float) rand() / RAND_MAX;

    /* Scale the random value to fit between 50 to 75 */
    t_n = temp * 25 + 50;
```

```
      /* Calculate t_RTT and d_RTT */
      t_rtt_new = (alpha * t_rtt_old) + ((1 - alpha) * t_n);
      d_rtt_new = (beta * d_rtt_old) + ((1 - beta) * fabs (t_n - t_rtt_old));

      if (t_rtt_new < t_rtt_min)
        t_rtt_min = t_rtt_new;
      if (t_rtt_new > t_rtt_max)
        t_rtt_max = t_rtt_new;

      if (d_rtt_new < d_rtt_min)
        d_rtt_min = d_rtt_new;
      if (d_rtt_new > d_rtt_max)
        d_rtt_max = d_rtt_new;

      t_rtt_sum += t_rtt_new;
      d_rtt_sum += d_rtt_new;

      printf ("t_RTT: %f d_RTT: %f\n", t_rtt_new, d_rtt_new);
      t_rtt_old = t_rtt_new;
      d_rtt_old = d_rtt_new;
    }
    printf ("t_RTT min: %f t_RTT max: %f t_RTT avg: %f\n",
             t_rtt_min, t_rtt_max, (t_rtt_sum / 500.0));
    printf ("d_RTT min: %f d_RTT max: %f d_RTT avg: %f\n",
             d_rtt_min, d_rtt_max, (d_rtt_sum / 500.0));
}
```

According to the preceding program the average value of t_{RTT} = 61.6924 and d_{RTT} = 7.1139. These values are averaged from a sample of over 500 values.

48. Suppose that the advertised window is 1 Mbyte long. If a sequence number is selected at random from the entire sequence number space, what is the probability that the sequence number falls inside the advertised window?

Solution:

If the sequence number field is 32 bits in length and the advertised window is 1Mbyte long, the probability that the sequence number falls inside the advertised window is:

$$P = (1 \times 10^6) / 2^{32} = 2.33 \times 10^{-4}$$

49. Explain the relationship between advertised window size, RTT, delay-bandwidth product, and the maximum achievable throughput in TCP.

Solutions follow questions:

a. Plot the maximum achievable throughput versus delay-bandwidth product for an advertised window size of 65,535 bytes.

First consider delay-bandwidth product, $DBP = R*2t_p$. Here delay $2t_p$ is the propagation time that elapses from when a bit is sent by a source to the destination to when the bit is returned back to the source. This is the minimum time that elapses from when a packet leaves a source to when the acknowledgment is received. The delay-bandwidth product *DBP* is then the number of bits (or bytes) that are in the network from when the source transmits continuously at the maximum rate to when the bits return immediately back to the source.

The round-trip time *RTT* is the time that actually elapses from when a packet is sent to when its acknowledgment is received. *RTT* includes not only the propagation delay, but also queueing and processing delays. The advertised window W places a limit on the amount of information that a source can have outstanding in the network.

Consider the time from when a byte leaves the source to when its acknowledgment is received (that is, consider a *RTT*). In that time, the source will have transmitted at most a window-full of bytes into the network. Therefore the window size divided by the *RTT* places a limit on the throughput *r*, that is, the rate at which information can be transmitted into the network: $r < W/RTT$.

The throughput cannot exceed the maximum bit rate $R = DBP/2t_p$ that is available for the source to transmit into the network. Therefore, the throughput increases as the window size is increased, but cannot exceed the bit rate *R*:

$$\text{Throughput} = r = \min\{R, W/RTT\} = \min\{DBP/2*t_p, W/RTT\}$$

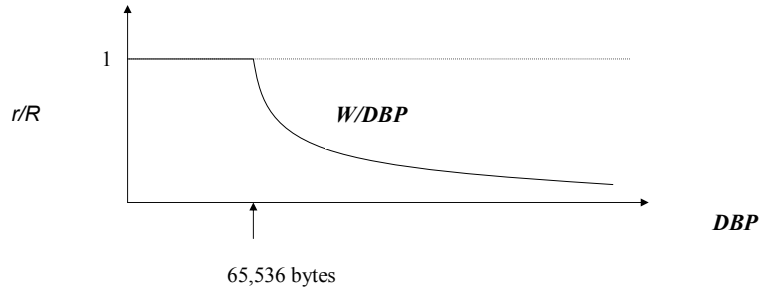
Suppose that the window size is less than the delay-bandwidth product. We then expect that the source cannot transmit at the maximum bit rate *R*. Indeed, we have that:

$$r < W/RTT < W/2t_p.$$

Therefore we have that:

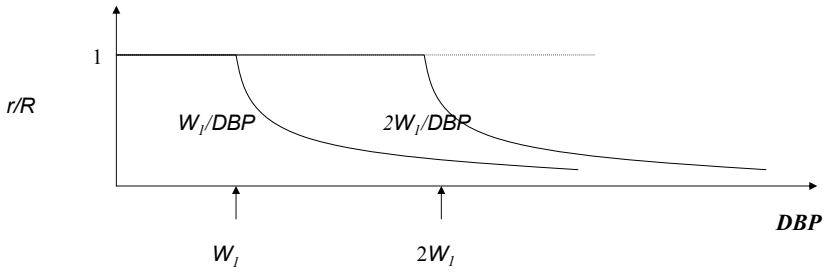
$$r/R < W/(R*2t_p) = W/DBP.$$

We conclude that the ratio of the maximum achievable throughput to *R* is less than the ratio of the window size to the *DBP*, as shown in the figure below.



b. In the above plot include the maximum achievable throughput when the above window size is scaled up by a factor of 2^K, where $K = 4, 8, 12$.

The following figure shows the case where the window size is doubled.



c. Place the following scenarios in the plot obtained in part (b): Ethernet with 1 Gbps and distance 100 meters; 2.4 Gbps and distance of 6000 km; satellite link with speed of 45 Mbps and RTT of 500 ms; 40 Gbps link with distance of 6000 km.

The window sizes (in bytes) of interest are: a. 65,536=2^{16}; b. $2^{20}=10^6$; c. $2^{24}=16.7x10^6$; d. $2^{28}=2.68x10^8$.

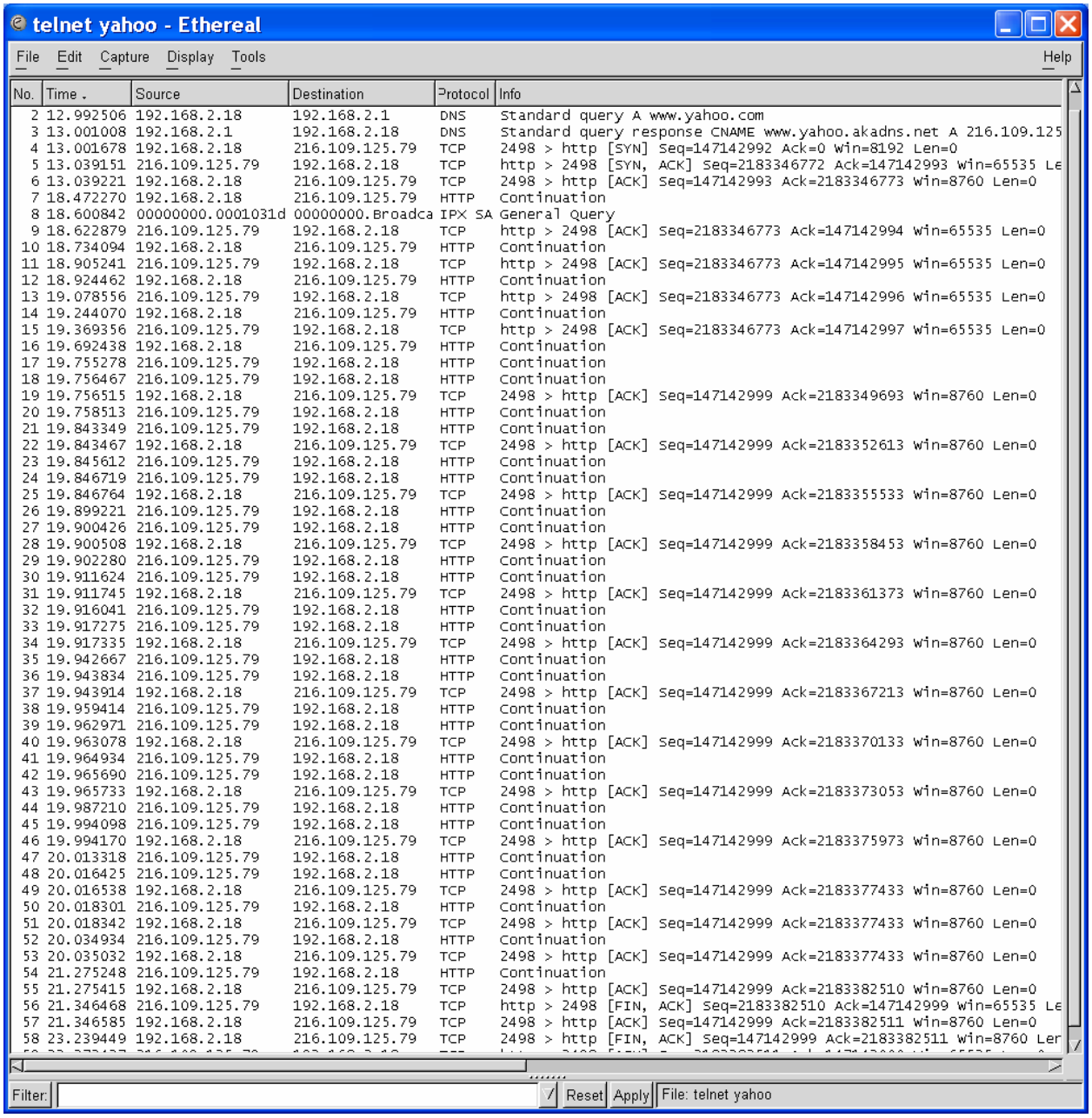
Case	*DBP* implied
Ethernet *R* = 1 Gbps *D* = 100 m	$T_p = 100 / 2.5x10^8 = 4x10^{-7}$ $DBP = 2*T_p*R = 8x10^{-7} * 1x10^6$ $DBP = 8 \times 10^{-1}$ ***DBP* < 1 bit** ***Scenario a***
OC-48 Link *R* = 2.4 Gbps *D* = 6000 km	$T_p = 6 \times 10^3 / 2.5x10^5 = 2.4 \times 10^{-1}$ $DBP = 2.4 x10^{-1} * 2.4x10^9$ $DBP = 2.4 \times 10^8$ ***DBP* = 2400 Mbits (300 Mbytes)** ***Scenario slightly beyond d***
Satellite link *R* = 45 Mbps *RTT* = 500 ms ($5x10^{-1}$ sec)	*DBP* = *RTT*R* $DBP = 5x10^{-1} * 45x10^6$ $DBP = 225 \times 10^5$ bits ***DBP* = 21 Mbits (< 2.6 Mbytes)** ***Scenario c***
OC-768 link *R* = 40 Gbps *D* = 6000 km ($6x10^6$ m)	$T_p = 6 \times 10^3 / 2.5x10^5$ $T_p = 2.4 \times 10^{-1}$ $DBP = 2*T_p*R$ $DBP = 2*2.4 x10^{-1} * 40x10^9$ $DBP = 192 \times 10^8$ ***DBP* = 18000 Mbits (2288 Mbytes)** ***Scenario beyond d***

50. Use a network analyzer to capture the sequence of packets in a TCP connection. Analyze the contents of the segments that open and close the TCP connection. Estimate the rate at which information is transferred by examining the frame times and the TCP sequence numbers. Do the advertised windows change during the course of the connection?

Solution:

The packet capture below is from a session with the HTTP server for yahoo.com. The initial sequence number for segments from the server can be seen from frame 5 to be 2183346772, so the first data byte has sequence number 2183346773. The FIN sent from the server in frame 56has sequence number 2183382510. Thus the server sent 35, 737 bytes. These bytes were received in the period between frame 7 (time=18.47 seconds) and frame 54 (time =21.27 seconds), so the elapsed time was about 2.8 seconds. The bit rate is then about 101 kbps.

The client window size remains fixed at 8760 bytes and the server window size remains fixed at 65535 bytes.



51. Explain why framing information is required even in the case where frames are of constant length.

Solution:

To detect frame boundaries based on the frame length, the first frame must be detected correctly, which requires synchronization information. Also the receiver must have exact timing of the transmitter which is not practical and as a result the bit synchronization may be lost. The synchronization may be lost because of bit errors or loss of data, or loss of signal in the link.

52. Perform the bit stuffing procedure for the following binary sequence: 1101111111011111110101.

Solution:

The inserted stuff bits are underlined.

1101111111011111110101 → 11011111<u>0</u>11011111<u>0</u>110101

53. Perform bit de-stuffing for the following sequence: 1110111110111111001111110.

Solution:

The removed stuff bits are indicated by a '-'.

1110111110111111001111110 → 111011111-11111-011111-

54. Consider the PPP byte stuffing method. What are the contents of the following received sequence of bytes after byte destuffing:

0x7D 0x5E 0xFE 0x24 0x7D 0x5D 0x7D 0x5D 0x62 0x7D 0x5E

Solution:

<u>0x7D 0x5E</u> 0xFE 0x24 <u>0x7D 0x5D</u> <u>0x7D 0x5D</u> 0x62 <u>0x7D 0x5E</u>

→ 0x7E 0xFE 0x24 0x7D 0x7D 0x62 0x7E

55. Suppose that GFP operates over an error-free octet-synchronous physical layer. Find the average time the GFP receiver spends in the hunt state.

Solution:

At the beginning of its operation, the GFP receiver takes 4 bytes and checks to see if the second two bytes correspond to the check sum of the first pair of bytes. There is exactly one 16-bit pattern that corresponds to this checksum and so the probability that this pattern occurs at random is 2^{-16} $=1.5\text{x}10^{-5}$, so most of the time the receiver immediately determines a given phase is incorrect. If the frame size is F, then on average F/2 bytes will be examined until the beginning of the frame is identified.

56. For GFP framing find the probability that the PLI and cHEC are not consistent. Assume that bit errors occur at random with probability p. Find the probability that the PLI and cHEC are not consistent in two consecutive frames.

Solution:

Considering that single bit errors are corrected:

$$P_{error} \approx P[errors \geq 2] \approx \binom{32}{2} p^2 (1-p)^{30} \approx 496 p^2$$

Probability of error in two consecutive frames = $(P_{error})^2 \approx 246016 p^2$

For example, if $p = 10^{-6}$, then probability of error in two consecutive frames is approximately 2×10^{-7}.

57. What is the maximum efficiency of GFP with respect to header overhead?

Solution:

The header is at least 4 bytes and the payload is at most 2^{16} bytes.

Maximum efficiency = $2^{16} / (2^{16} + 4) = 0.99994$

58. Can GFP be used to carry video signals? If so, how would you handle the case where each video image produces a variable length of compressed information? How would you handle the case where each video frame produces a fixed length of compressed information?

Solution:

GFP can be used to carry video signals. In the case of variable-length information, frame-mapped mode is used in which variable-length GFP frames are sent. In the case of fixed-length information, transparent mode is used in which fixed-length GFP frames are sent.

59. Suppose a server has a 1 Gbps Ethernet link to a GFP transmitter. The GFP has an STS-1 SONET connection to another site. The GFP transmitter has a limited amount of buffering to accommodate frames prior to transmission. Explain how the GFP transmitter may use flow control on the Ethernet link to prevent buffer overflow. What type of flow control is preferred if the server and GFP transmitter are co-located? If the server and transmitter are 5 km apart?

Solution:

The GFP transmitter sets two threshold levels in the buffer for low and high buffer fill levels. Once the buffer fill level exceeds the high buffer fill threshold, the flow control mechanism stops the server. Once the buffer fill level crosses the low buffer fill threshold, the flow control mechanism signals the server to start transmission.

If the server and GFP transmitter are co-located X-ON X-OFF is preferred. If the server and GFP transmitter are not co-located sliding-window flow control is preferred.

60. An inverse multiplexer combines n digital transmission lines of bit rate *R* bps and makes them appear as a single transmission line of *n* x *R* bps. Consider an inverse multiplexer that combine multiple STS-1 SONET lines. Find an appropriate value of *n* to carry a 1 Gbps Ethernet stream. What is the improvement in efficiency relative to carrying the Ethernet stream on an OC-48 link?

Solution:

$1 \times 10^9 / 50 \times 10^6 = 20$, Efficiency = 100%

Efficiency if OC-48 is used: $1x10^9 / 2.4x10^9 = 42\%$

Improvement = 100 - 42 = 58%

61. Suppose that a 1-Megabyte message is sent over a serial link using TCP over IP over PPP. If the speed of the line is 56 kbps and the maximum PPP payload is 500 bytes, how long does it take to send the message?

Solution:

Assume the overhead in one packet is equal to 8 bytes for the PPP header plus 20 bytes for the IPv4 header and 20 bytes for the TCP header. Thus, the total overhead in bits is 8 x (8 + 20 + 20) = 384 bits. Thus,

Time to send 8×10^5 bits = (time for 1 packet)(# of packets needed)

$$= (\frac{n_f}{R})(\frac{8\times10^6}{n_f - n_o})$$

$$= (\frac{8\times500 bits}{56\times10^3 bps})[\frac{10^6 bytes}{(500-70) bytes}]$$

$$= 0.07143\frac{\text{sec}}{packet}\times 2326\, packets$$

= 166.14 seconds

62. Compare PPP and GFP. Identify situations where one is preferable to the other.

Solution:

PPP uses character-based framing which requires byte stuffing. As a result the frame size varies depending on the character pattern. GFP resolves this problem by using frame length based framing. GFP can support synchronous services, while PPP cannot. However PPP provides useful capabilities through link and network control protocols.

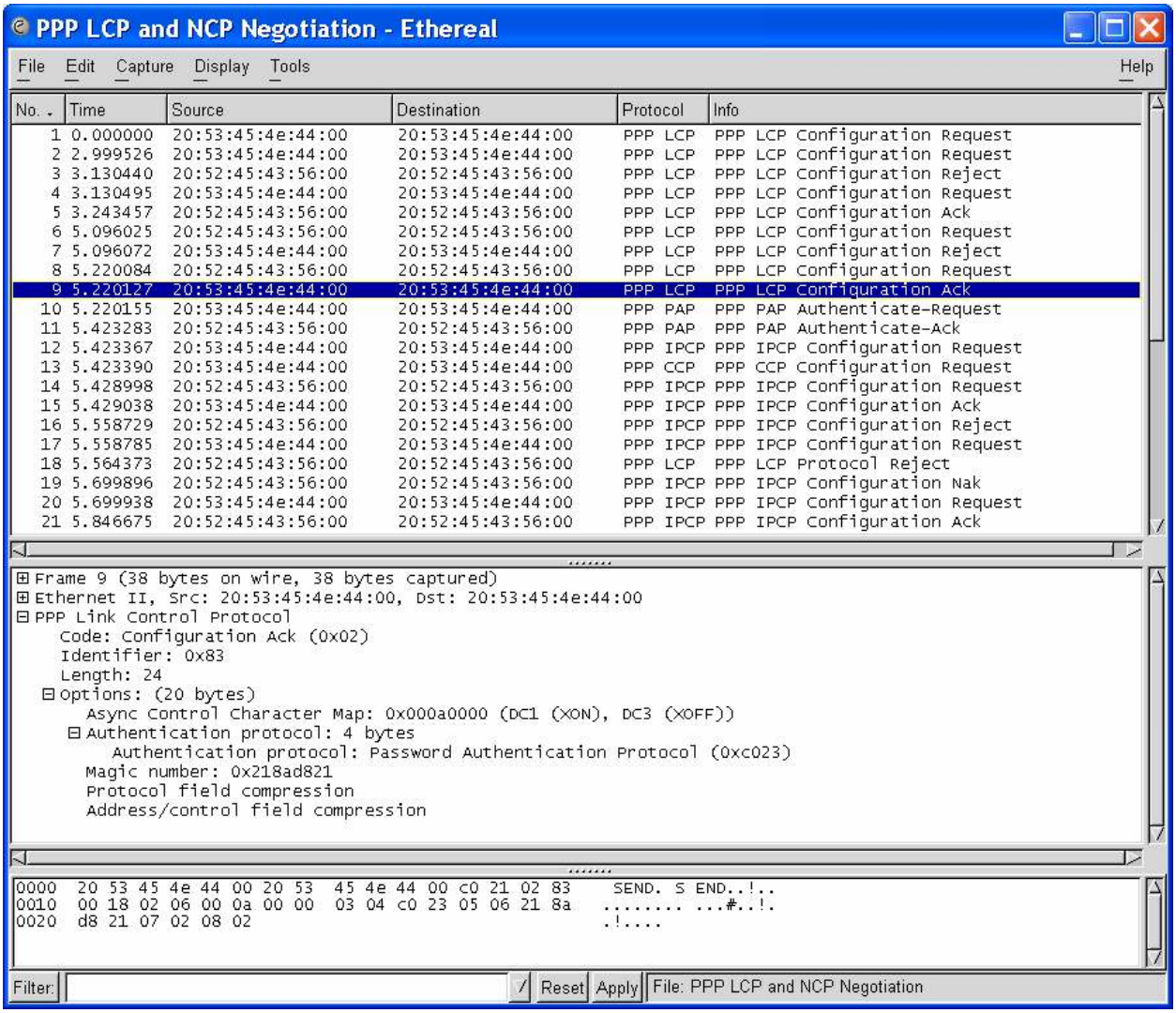
63. Use the Ethereal packet capture tool to analyze the exchange of packets during PPP setup using ad dial up connection to a local ISP. Start the Ethereal packet capture and then initiate the connection to the ISP. The number of captured packets will stop increasing after some point.

Solutions follow questions:

a. Analyze the options fields of the packets during the LCP negotiations. You may need to consult RFC 1662 to interpret some of the packet contents.

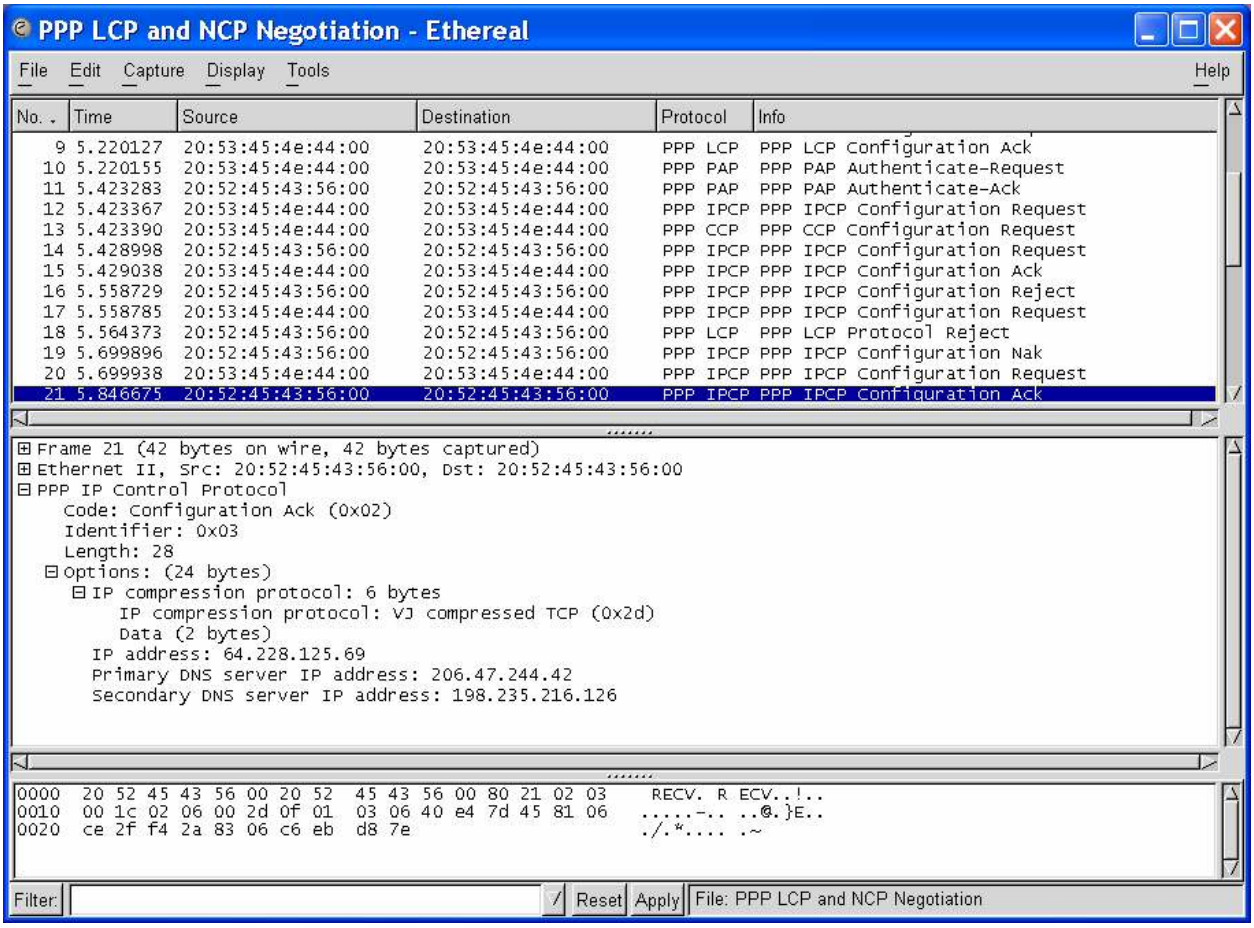
b. Examine the packet contents during the PAP exchange. Repeat the packet capture using CHAP.

The screen capture below shows a sequence of LCP and NCP negotiations for PPP. The twp stations exchange LCP messages where they propose and reject various configuration options. The highlighted frame (9) shows the final ACK accepting a configuration. It can be seen from the middle pane that PAP authentication will be used. Frame 10 sends the password request, and frame 11 sends the password.



c. Analyze the IPCP packet exchange. What IP addresses are configured during the exchange? What other options are negotiated? You may wish to consult RFC 1332 for this part.

The screen capture below shows a sequence the final IPCP ACK message confirming the use of protocol field compression and various IP addresses.



64. Explain the differences between PPP and HDLC.

Solution:

HDLC can support various data transfer modes, supports multipoint links and point to point links, and can implement error control and flow control mechanisms. PPP uses HDLC-like frames but does not use error control and flow control protocols. Instead PPP supports powerful link and network control protocols.

PPP is character based and can be implemented on any physical layer, HDLC is bit based and can be implemented only on bit synchronous physical layer.

65. A 1.5 Mbps communications link is to use HDLC to transmit information to the moon. What is the smallest possible frame size that allows continuous transmission? The distance between earth and the moon is approximately 375,000 km, and the speed of light is 3 x 10^8 meters/second.

Solution:

The round trip propagation delay is:

$$2t_{prop} = 2\frac{(375\times10^6\, m)}{3\times10^8\, m/s} = 2.50\,\text{sec}$$

To allow for continuous transmission, we must use Go-Back-N or Selective Repeat.

Go-Back-N:

If $N = 7$ → $\frac{7n_f}{1.5Mbps} = 2.5s$ → n_f = 535 715 bits

If $N = 127$ → $\frac{127n_f}{1.5Mbps} = 2.5s$ → n_f = 29 528 bits

Selective Repeat:

If $N = 4$ → $\frac{4n_f}{1.5Mbps} = 2.5s$ → n_f = 973 500 bits

If $N = 64$ → $\frac{64n_f}{1.5Mbps} = 2.5s$ → n_f = 58 594 bits

66. Suppose HDLC is used over a 1.5 Mbps geostationary satellite link. Suppose that 250-byte frames are used in the data link control. What is the maximum rate at which information can be transmitted over the link?

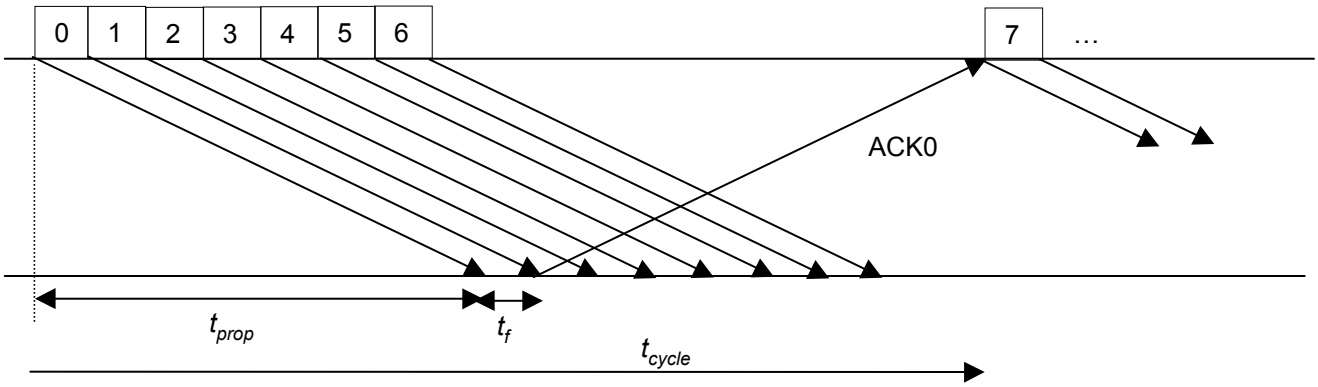
Solution:

R = 1.5 Mbps, and n_f =250 bytes or 2000 bits (250 x 8). The distance that the information must travel is the earth-to-satellite distance, or $d \approx$ 36,000 km. The speed of light c is 3 x 10^8. We can calculate the propagation delay and processing rate as follows:

$t_{prop} = d/c$ = 36 x 10^6 / 3 x 10^8 = 120 ms

$t_f = n_f/R$ = 2000/1.5 x 10^6 = 1.33 ms

We can use either Go-Back-N or Selective Repeat ARQ. The default window size is N = 7 (with a 3-bit sequence number).



The maximum information rate is achieved with no error, and hence, no retransmission.

t_{cycle} = minimum time to transmit a group of N packets

= $t_f + 2\ t_{prop}$ = 1.33 + 2x120 = 241.33 ms

n = no. of bits transmitted in a cycle = $N.n_f$ = 7x2000 = 14,000 bits

R_{max} = no. of bits sent in a cycle / minimum cycle time = n/t_{cycle} = 58 kbps

If the extended sequence numbering option (7-bit) is used, the maximum send window size would be $N = 2^7 - 1 = 127$, and hence, the maximum information rate is:

$R_{max} = N.n_f / t_{cycle}$ = 127x2000/(241.33x10^{-3}) = 1.052 Mbps

67. In HDLC how does a station know if a received frame with the fifth bit set to 1 is a P or an F bit?

Solution:

If the station is a secondary station, the bit is a 'P' (and the station is being *Polled* for more frames). If it is a primary station, the bit is a 'F' bit (indicating the *Final* frame of the current transmission).

68. Which of the following statements are incorrect?

Solutions follow questions:

For this question, one must just keep in mind that all frames always contain the address of the secondary station.

a. A transmitting station puts its own address in command frames.

Incorrect. Command frames are destined for secondary stations, so this transmitter would put the address of the secondary station in the frame.

b. A receiving station sees its own address in a response frame.

Incorrect. Responses come from secondary stations to primary stations. Thus the address in the frame is that of the sender in this case.

c. A response frame contains the address of the sending station.

TRUE. Response frame originate in secondary stations, so they will have contain the address of the sender.

69. In HDLC suppose that a frame with P = 1 has been sent. Explain why the sender should not send another frame with P = 1. What should be done to deal with the case where the frame is lost during transmission?

Solution:

The poll bit is like a token that is passed to a secondary so it can transmit. At the end of transmission the secondary must return it in the form of final bit.

If the poll frame is lost then the token no longer exists and no station can transmit. The server must create a new token (poll frame) only after it is sure that the previous poll frame is lost and not just delayed. Consequently it must wait for certain period of time.

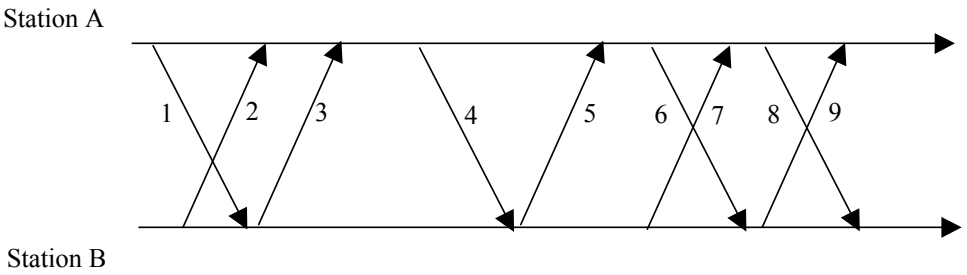
70. HDLC specifies that the N(R) in a SREJ frame requests the retransmission of frame N(R) and also acknowledges all frames up to N(R) – 1. Explain why only one SREJ frame can be outstanding at a given time.

Solution:

Suppose two outstanding SREJ frames exist. Let frame A have N(R) = m and frame B have N(R) = n. Without loss of generality, suppose $n > m$.

Since each SREJ frame with value N(R) implicitly acknowledges all previous frames up to N(R) – 1, frame A indicates that frame *m* has not yet been received and frame B indicates that frame *m* has been received. Thus, if two SREJ are allowed to be outstanding at the same time, contradictory information will be sent to the receiver.

71. The following corresponds to an HDLC ABM frame exchange with no errors.



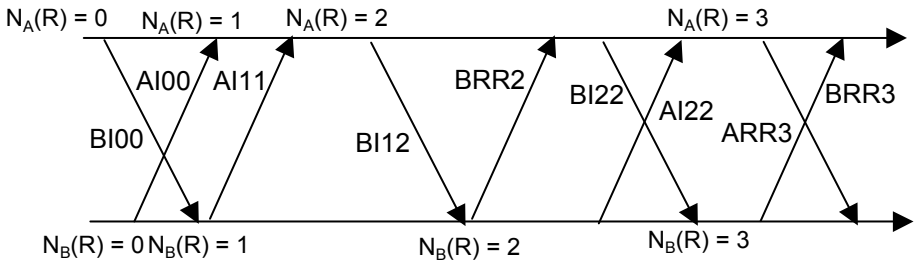
1. BI00 2. AI00 3. xIxx 4. xIxx 5. xRRxx

6. xIxx 7. xIxx 8. xRRxx 9. xRRxx

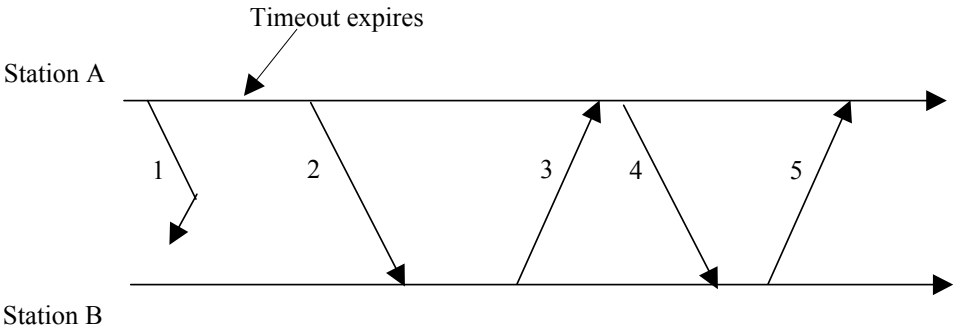
a. Complete the diagram by completing the labeling of the frame exchanges.

b. Write the sequence of state variables at the two stations as each event takes place.

Solution:



72. Assume station B is awaiting frame 2 from station A.

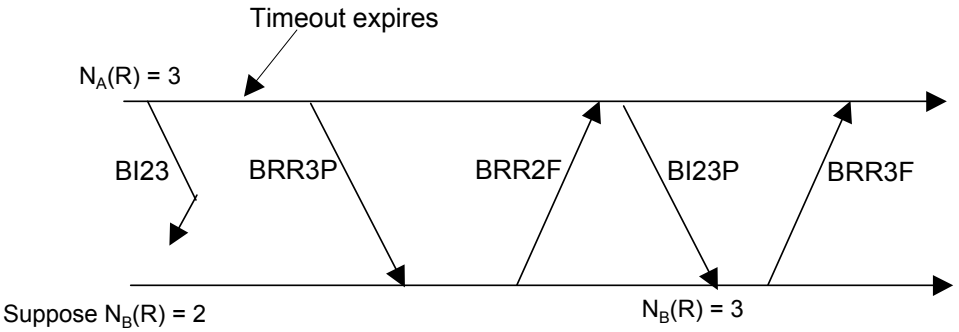


1. BI23 2. xRRxP 3. xRRxy 4. xIxx 5. xRRx

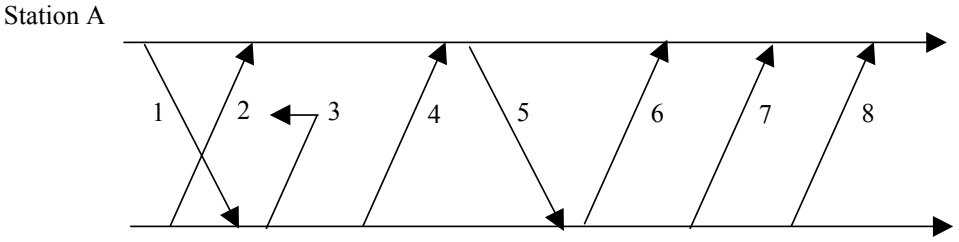
a. Complete the diagram in HDLC ABM by completing the labeling of the frame exchanges.

b. Write the sequence of state variables at the two stations as each event takes place.

Solution:



73. The following corresponds to an HDLC ABM frame exchange.



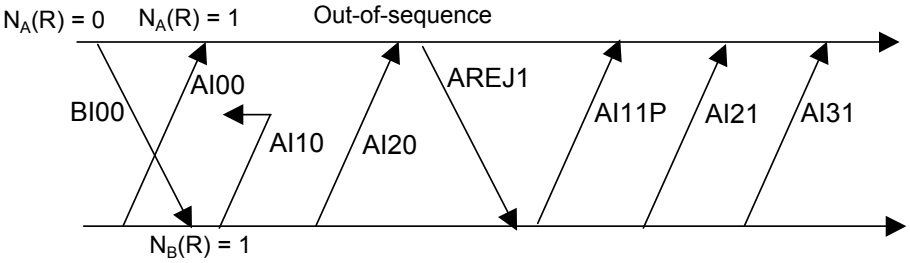
1. BI00 2. AI00 3. xIxx 4. xIxx

5. xREJx 6. xIyx 7. xyxx 8. xyxx

a. Complete the diagram by completing the labeling of the frame exchanges.

b. Write the sequence of state variables at the two stations as each event takes place.

Solution:



74. Suppose that packets arrive from various sources to a statistical multiplexer that transmits the packets over a 64 kbps PPP link. Suppose that the PPP frames have lengths that follow an exponential distribution with mean 1000 bytes and that the multiplexer can hold up to 100 packets at a time. Plot the average packet delay as a function of the packet arrival rate.

Solution:

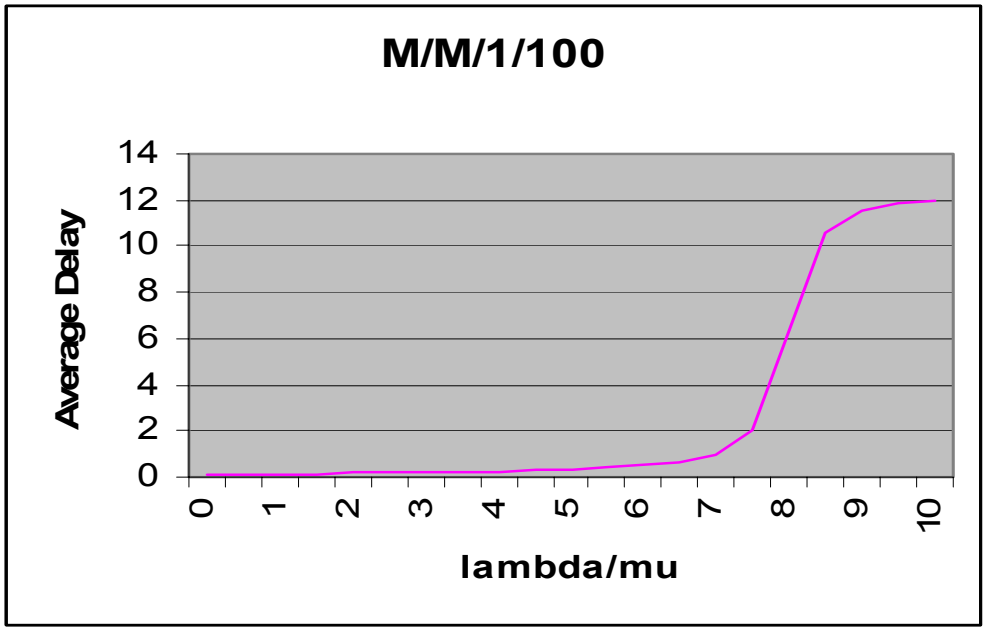
This is an M/M/1/K system in which:

Service rate μ = 64000 bps / 8000 bits/packet = 8 packets/sec
Packet length *E[L]* = 8000 bits
Buffer size *K* = 100 packets

Thus:

$$E[T] = \frac{E[N]}{\lambda(1-P_L)}$$

where $P_L = \dfrac{(1-\frac{\lambda}{\mu})(\frac{\lambda}{\mu})^K}{1-(\frac{\lambda}{\mu})^{K+1}}$ and $E[N] = \dfrac{\lambda}{\mu-\lambda} - \dfrac{(K+1)\lambda^{K+1}}{\mu^{K+1}-\lambda^{K+1}}$



75. Suppose that the traffic that is directed to a statistical multiplexer is controlled so that ρ is always less than 80%. Suppose that packet arrivals are modeled by a Poisson process and that packet lengths are modeled by an exponential distribution. Find the minimum number of packet buffers required to attain a packet loss probability of 10^{-3} or less.

Solution:

$$P_{loss} = \frac{(1-\rho)\,\rho^k}{1-\rho^{k+1}}$$

with *k* = 24 P_{loss}= 0.000948

76. Suppose that packets arrive from various sources to a statistical multiplexer that transmits the packets over a 1 Mbps PPP link. Suppose that the PPP frames have constant length of *L* bytes and that the multiplexer can hold a very large number of packets at a time. Assume that each PPP frame contains a PPP, IP, and TCP header in addition

to the user data. Plot the average packet delay as a function of the rate at which user information is transmitted for L = 250 bytes, 500 bytes, and 1000 bytes.

Solution:

Assume the overhead in one packet is equal to 8 bytes for the PPP header plus 20 bytes for the IPv4 header and 20 bytes for the TCP header.

$R = 10^6$ bps
L = constant (250, 500, and 1000 bytes)
$K \rightarrow \infty$
n_o = 48 bytes
$\mu = R/(8*L)$ packets/sec = 500 (L=250 B); 250 (L=500 B); 125 (L=1000)

Because of the constant length packets, this is an M/D/1 system.

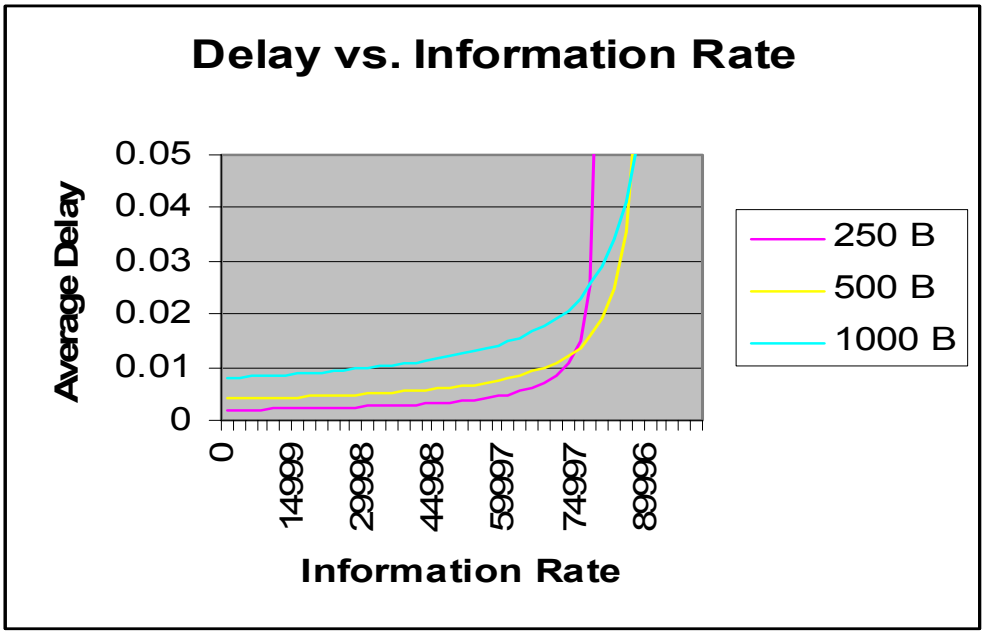
Average delay, $E[T_D] = (1 + \frac{\lambda}{2(\mu - \lambda)})\frac{1}{\mu}$

Let γ be the rate in bits per second at which user information transferred.

$$\gamma = \lambda(8L)\frac{(L - n_o)}{L} = \lambda(8L)(1 - \frac{48}{L}) \text{ or } \lambda = \frac{\gamma}{8L - 384}$$

And so, the delay as a function of the user information bit rate is

$$E[T_D] = (1 + \frac{\frac{\gamma}{8L - 384}}{2(\frac{R}{8L} - \frac{\gamma}{8L - 384})})\frac{8L}{R}$$



77. Suppose that a multiplexer receives constant-length packet from $N = 60$ data sources. Each data source has a probability $p = 0.1$ of having a packet in a given T-second period. Suppose that the multiplexer has one line in which it can transmit eight packets every T seconds. It also has a second line where it directs any packets that cannot be transmitted in the first line in a T-second period. Find the average number of packets that are transmitted on the first line and the average number of packets that are transmitted in the second line.

Solution:

The probability that there are k packet arrivals in a T-second period is given by the binomial distribution with parameters $N = 60$ and $p = 0.1$. The average number of arrivals is $Np = 6$. The average number of arrivals that get transferred to the first line is given by:

$$\sum_{k=0}^{8} k \binom{60}{k} (0.1)^k (0.9)^{60-k} = 4.59$$

The remainder of the packet arrivals is sent to the second line, so the average number sent to line 2 is 6 – 4.59 = 1.41 packets per T-second period.

78. Discuss the importance of queueing delays in multiplexers that operate at bit rates of 1 Gbps or higher.

Solution:

By aggregating traffic, better delay performance is achieved, as illustrated in section 5.5. In multiplexers that operate at bit rates of 1 Gbps or higher, the queuing delay experienced by packets is much shorter than it would be if many lower bit rate systems were used.

Changes to this document

Version No.	*Problem No.*	*Change*
1.00 to 1.01	5.14a	Changed reference Figure 5.12 to 5.10 in solution text
	5.35	Changed reference Table 5.12 to Equation 5.11 in solution text.
	Throughout	Cosmetic changes resulting in changes in pagination