Homework 1 - 4390 Summer 2018

Alex Lundin

Question 1:

=============

a) Consider a network in which packets are delivered reliable across each link. That is, for each pair of adjacent nodes A - B, the datalink layer has an ARQ protocol (such as stop-and-wait) that ensures packets sent by A do arrive at B. In this network, why do you think is necessary for the transport layer to also have its own ARQ protocol and do retransmissions of its packets?

**Answer: There is the possibility for link-failures on networks. So, if the entire data link layer fails, the transport layer will be able to detect this since it has it’s own ARQ protocol as a backup.**

b) For each of the following, identify which OSI layer is responsible for performing the described action:

i) Make sure that a packet is delivered reliably across a single physical link

**Data Link**

ii) Describes how a binary tree data structure is to be encoded inside the message

**Presentation**

iii) Makes sure that a message is delivered reliably from the source host to the destination host

**Network**

iv) Coordinates the transmission of multiple flows of data across multiple sources and multiple destinations (for example, a video conference)

**Transport**

v) Defines how the bits are converted into voltages

**Physical**

vi) Decides how a data packet is to be routed from one link to the next.

**Network**

Question 2 -

=============

a) Assume you have a link that has 50ms of round-trip-time, the transmission rate is 3Mbits/sec, and each packet is 1Kbyte. If we are using the concurrent logical channels protocol, how many logical channels do we need to maintain 100% throughput (assuming no message loss)

1.) Convert Transmission rate = 3 Mbits/s = 375,000 Bytes/s

2.) Calculate Bytes per cycle = Transmission rate \* rrt = 375,000 Bytes/s \* .05s = 18750 Bytes

3.) Calculate number of channels = Bytes per cycle / Packet size = **18.75 channels, so 19 channels**

b) Assume three of the above (i.e. part (a)) logical channels have lost a message and are stuck waiting for the timeout to occur (assume the timeout value unnecessarily large, way too big). How much is the reduction in throughput?

1.) 3 channels waiting / 19 total channels = .158

**2.) 16% reduction in throughput**

Question 3

===========

Consider the following sequence of data bits

1011

0100

1110

1010

0011

How would this be translated into physical bits if we use:

a) NRZ

red means transition

10110100111010100011

**1010101010101**

b) NRZI (high value initially)

red means transition

10110**1**00111010100011

**1010101**

c) Manchester

**1001101001100101101010011001100101011010**

d) 4B/5B

10111

01010

11100

10110

10101

**1011101010111001011010101**

Question 4:

===========

a) Consider bit-stuffing (also known as bit-insertion) the way I present it in the slides, NOT the way it is presented in the book. Consider k = 2. If the data bits of the frame to be sent are

1110011010110011101

what are the actual bits sent by the sender? Please include some idleness bits before and after the frame.

**Bold is data**

Red is added 0

111110**11010011001011000110101**01111111

b) Is it possible to have k=1? Why or why not?

**Yes it is possible because data is able to be represented this way with no problem. It does not seem efficient at all though, because the size of every message will be almost double of the original.**

Question 5

==========

a) Consider the parity bit protocol with the p's, q's, and the additional r bit. What is the hamming distance of this protocol? Briefly explain why

Hamming distance is the distance between two equal-length code words. This is calculated by XOR(p,q). But there is an extra r bit on each item, so add 1 to whatever the size of p and q end up being.

**XOR(p+1, q+1), assuming the r variable is only 1 bit added to the end of both p and q.**

b) Assume we have a CRC protocol that satisfies all the desirable properties that we described in the slides. What is the hamming distance of this protocol? Briefly explain why.

c) For both a) and b), can these protocols be used for error correction, and if so, how many bits can they correct? (i.e., can they perform x-bit correction, and if so, what is x?)

Question 6

==========

a) Assume G = 100101, and M = 10110011. What check bits are sent by the sender?

**101 for n = 4**

b) Argue that if G has at least three terms (e.g. G = 1011 or G = 1110 or G = 1001100), then it can catch all errors E of the form x^i + x^(i-1) (i.e., two consecutive bits).

Question 7

========

a) Assume that packet size is 1KB, the round-trip-time is 25ms, and the bandwidth of the channel is 1200KB/s. What is the throughput of stop-and-wait?

1.) Convert Transmission rate = 1200 KB/s

2.) Calculate Bytes per cycle = Transmission rate \* rrt = 1200 KB/s \* .25s = **3000 KBytes**

b) What is the minimum number of logical channels in the concurrent-logical channels protocol that are needed to ensure we maximize the throughput

3.) Calculate number of channels = Bytes per cycle / Packet size = 3000 Kbytes / 1 KB= **3 channels**

c) What if instead of using the number of logical channels you gave in b), we use twice that number. What will happen to the throughput? Briefly argue why.

**Throughput will not be affected for ideal conditions. The extra channels will be used when a channel is waiting on a response. These extra channels are a safety net for un-anticipated congestion.**

Question 8

=======

Assume we have the cumulative acknowledgment protocol with unbounded sequence numbers. Assume the channel can lose and reorder messages (but not duplicate). Furthermore, assume that the SWS = 1, and that the timeout of the sender is accurate,. i.e., the last action of the sender is replaced by

when timeout frame#ch.sender.receiver = 0 and ack#ch.receiver.sender = 0 and LAR < LFS then

send frame(LAR+1) to receiver.

Under these conditions, is the sequence number in the acknowledgment really necessary? Briefly argue why yes or why no.

**No the sequence number is not needed because the Sender Window Size (SWS) is size 1 on this cumulative acknowledgement setup. Meaning that we are only ever waiting on acknowledgement of a cumulative set of 1 message. So the sequence number won’t do anything for the troubleshooting. The only message that can be lost is the last message sent. No need for sequencing.**