CSE 123: Computer Networks

Homework 2

Out: 10/22 Due: 10/29

Instructions

1. *Turn in a physical copy at the beginning of the class on 10/29*
2. *Ensure the HW cover page has the following information clearly written*
    a. *Name*
    b. *UCSD email*
    c. *PID*

1. Sliding Window Protocol

Suppose you are designing a sliding window protocol for a 4-Mbps point-to-point link that is 9 x 10^4 km long. Assuming each frame carries 4 KB of data, what is the minimum number of bits you need for the frame sequence numbers if

L = 9 x 10^7 m
S = 4 KB
B = 4 Mbps

The SWS should have enough frames to hold the packets that are needed to completely fill the pipe. This capacity, can be found using the bandwidth-delay product.
In this approach, we estimate the bandwidth-delay product by the number of frames it takes to fill a pipe completely.
T_{delay} = L/c = 0.3 s
RTT = 0.6 s
B_{fps} = B/S = 123 fps (approx)
N_f = B_{fps} x RTT = 74 frames (approx)

Thus, SWS should hold at least 74 frames.

a. RWS = 1

   If RWS = 1 then the receiver can handle only one frame at a time and hence to fully utilize the capacity of the link we'll need to have at least 74 unique frame sequence numbers.
   Thus, we'll need **7 bits.**

b. SWS = RWS

   In this case, the sender can send twice as many frames since the receiver can now store frames as well. Thus we can send as many as 148 frames to fully utilize the link.

Thus, in this case we'll need **8 bits.**

c. If we use the Stop-and-Wait ARQ protocol, what would be the maximum efficiency (defined as *frames in transit/maximum possible frames in transit)* that we can achieve?

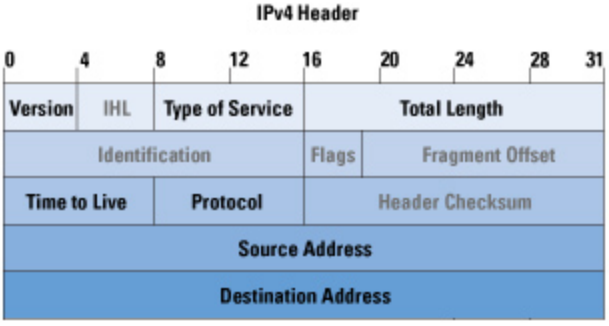
In Stop-and-Wait ARQ protocol we can have at most 1 frame in transit. However, we calculated that at the maximum we can have 74 frames in trasit.
Efficiency = 1/74 = 0.014 or 1.4%

Assume that communication happens at the speed of light, that is 3 x 10^8 m/s

2. IP Checksum and Endianness

The following IPv4 header, show in hex below, is received for an IP packet at its destination. Refer to the IPv4 header diagram below.

4500 003c 1c46 4000 4017 c311 aca8 0101 aca8 0102



a. What is the header checksum? (You can find it by mapping the hex values to the IPv4 header diagram)

Header Checksum = c311

b. Using the Internet checksum algorithm, determine if there were any errors in the transmission. (Show your work)

```
   4500
+  003c
+  1c46
+  4000
+  4017
+  c311
+  aca8
+  0101
+  aca8
+  0102
```

```
_____________
0002 FFFD
FFFD + 0002 = FFFF
~FFFF = 0000 (Taking 1's complement)
As the final sum is 0, we know that there were no errors
in the transmission.
```

c. What would this packet which is network-byte order look like when stored in the memory of a big-endian system?

The big-endian system is similar to the network-byte order.

#	Data
0	45 00 00 3C
4	1C 46 40 00
8	40 17 C3 11
12	AC A8 01 01
16	AC A8 01 02

d. How about in the memory of a little-endian system?

The little-endian system stores the least significant byte in the smallest address.

#	Data
0	3C 00 00 45
4	00 40 46 1C
8	11 C3 17 40
12	01 01 A8 AC
16	02 01 A8 AC

The question aimed at pointing out some of the problems that could surface when you store data in a different endian system.

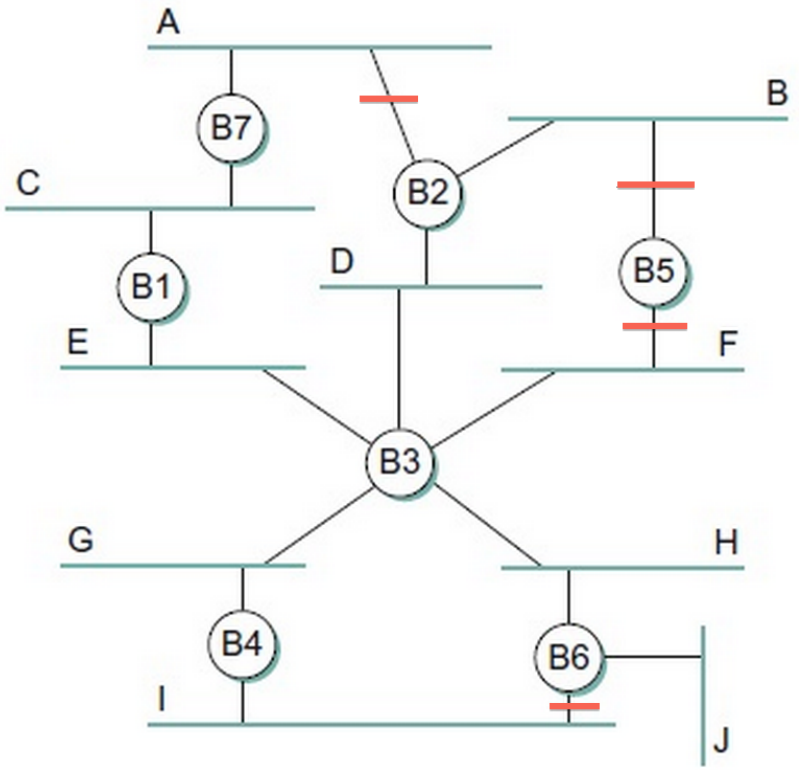
3. CRC

Suppose we want to transmit a message 1101 1001 0101 1001 and protect it from errors using the CRC generator $x^8 + x^6 + x^3 + 1$

a. What is the CRC generator sequence? (The CRC generator polynomial represented in a bit sequence)
   `CRC Generator Sequence: 101001001`
b. How many bits will the resulting frame check sequence be?
   `8 bits.`
c. What is the transmitted bit sequence (show your work)?
   `Transmitted Sequence: 1101 1001 0101 1001 1101 0000`
d. How large a burst of errors can be detected?
   `A burst error of maximum 8 bits can be detected.`

4. Spanning Tree Question

Given the extended LAN shown below, indicate which ports are not selected by the spanning tree algorithm.



5. Sliding Window Timeline

Consider a network where the sliding window protocol is in use with SWS = RWS = 2 frames and a one-way delay of 100 ms (i.e. for a frame sent at time t, it arrives at t + 100 ms). Assume that when multiple frames are all able to be sent according to the window

size, that they are sent 20 ms apart (i.e., frame 1 starts at time t, and frame 2 starts at time t + 20 ms).

a. What would be a reasonable timeout value for this link? Why not something smaller? How about larger?

A reasonable timeout would be anything slightly larger than the RTT. You ideally, want to have time for the packet to travel to the receiver and back while allowing some time for processing.

If the timeout value were smaller then we would be unnecessarily retransmit packets. If the value were much larger than we would be wasting time by unnecessarily over waiting.

b. Now, assume that by sending a frame every 5 ms we can saturate the link. In this case, determine the smallest SWS and RWS that maximizes throughput (i.e., keeps the link fully utilized).

The bandwidth-delay product indicates the capacity of the link. If the sender can send a frame every 5 ms, then the sender can send 40 frames in 1 RTT.

Thus, if we have SWS = RWS = 40 we'll be able to keep the link fully utilized.

6. 2D Parity

a. Explain briefly why 2D parity can detect all 1 bit errors. Can the 1bit errors be corrected?

A 2D parity is a combination of a row and column 1D parity. Hence, all 1-bit errors will be caught both in the row and column parity. This will help us identify the location of the 1-bit error and correct it.

b. Again, explain briefly why 2D parity can detect all 2 bit and 3 bit errors?

For 2-bit and 3-bit errors, either the row or the column parity will catch it. In the case, of 2-bit if the errors are in the same row then the column parity will catch it. Same goes, with 3-bit errors, that is, one of the bit-errors will lead to an incorrect parity and hence be detected.

c. Why can it not detect all 4 bit errors? Give an example of a 4 bit error that it can detect, and an example of a 4-bit error that it cannot detect.

2D parity can detect most of the 4-bit errors. However, if the 4 bit errors are grouped such that any of the bit errors has a bit error both in the same row and same column as it is, then in that case both the row and column parity will check out and hence not detect the errors.

ORIGINAL									4 BIT-ERRORS								
1	0	0	1	0	1	1	0	0	1	0	0	0	0	1	0	0	0
0	0	1	0	1	1	0	0	1	0	0	1	0	1	1	0	0	1
0	1	0	1	1	0	0	0	1	0	1	0	0	1	0	1	0	1
1	1	1	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0

Example of type of 4 bit errors that cannot be detected