Reading

Peterson and Davie, Chapters 3, 4, and 5.1-5.2.

Objectives

The purpose of this assignment is to reinforce your understanding of bridging, switching, Internet Protocol, and distributed routing algorithms.

Review Problem X. The required relationship among send window size, receive window size, and size of the sequence number space is $SWS + RWS \leq N$, where $N$ is the size of the sequence number space. However, this condition is sufficient only when messages cannot be reordered in the channel.

a. Give an example showing how the sliding window protocol can fail with $SWS + RWS = N$, if messages can be arbitrarily reordered in the channel, i.e. if a message can arrive after any number of messages that were transmitted after it. (Hint: $N=2$ is sufficient to demonstrate the problem.)

b. IP, of course, can reorder messages. Yet, as we shall see, TCP is a sliding window protocol. What relationship must hold among $SWS$, $RWS$, and $N$, and what constraint must be on the channel, in order to make the sliding window protocol work when the channel can reorder messages?

Review Problem Y. Consider a sliding window protocol with $SWS = RWS = 4$. The transmission rate is 100 Mbps, with a round-trip time of 10 milliseconds ($10^{-3}$ sec). Every frame is 12,500 bytes long. As usual, assume zero processing time at the receiver and the ack channel has infinite bandwidth, and the header has enough bits for sequence numbers to “keep the pipe full”.

a. Suppose the loss rate for frames or acks is 10%, i.e. with probability 0.1, either the frame or its ack is lost. Which provides the greatest increase in normalized throughput (i.e. efficiency of the protocol): doubling the transmission rate, halving the round-trip time, or doubling the frame size?

b. How does your answer change if the protocol has only 3 bits for sequence numbers in the header?

Problem A. Memory is getting cheaper all the time. It has been suggested that, instead of implementing IP forwarding by doing longest-prefix matching on forwarding table entries, that each router keep a single table with $2^{32}$ entries, containing the route for every possible IP address. This would enable forwarding lookup to be accomplished with one memory access. Assuming memory is indeed cheap enough to make this cost-effective, is this a good scheme? If not, what problem(s) do you foresee?
**Problem B.** Simulate the spanning tree algorithm for the network shown in Figure 3.37 in your textbook. Represent the network as a **bipartite graph** in which one set of vertices represent the bridges and the other represents the LANs (see figure below). Next to each bridge, write the triple it is currently broadcasting, i.e. (who I think is root, distance to root, my id). The initial values, where each bridge thinks it is root, are shown in the figure below. Next to each LAN, write the **minimum** (in lexicographic or “alphabetical” order) value of triple transmitted on that LAN. Again, the result of the first round is shown in the figure below.

Reproduce the graph, showing the evolution of the bridges’ tuples until the algorithm converges. You may assume the bridges all update their tuples at the same time. Indicate which interfaces are not selected at convergence. (Note: each edge in the bipartite graph represents a port.)

**Chapter 3 Exercises.** Do exercises 14, 15, 22, 33, and 36 at the end of Chapter 3.

**Chapter 4 Exercises.** Do exercises 7, 8, 10, 15, 20, 25, 29, 30, 32, 37, 38, and 52.

**Problem C.** [Note: You will need to refer to the OSPF specification (RFC 2328) to do this problem.] Consider Figure 4.32 in your text. First, assign IP addresses to each network and interface in the figure. Then, assuming OSPF is in use in the network, show the information contained in the Link-state announcements originated by Router R1 on each of its interfaces. Note that R1 has interfaces belonging to Areas 0, 1 and 2; you may assume all link costs are 1.