EECS 461 Problem Set #2
Due Date: Tuesday, December 8, 5pm

This problem set will be weighted one third of the total number of points you missed (or will miss) on the midterm and final, or roughly 1% per question. The intent is that the problem set will be optional to those people who did well on the midterm, and strongly encouraged for those students who scored lower than the mean. For people in the middle, our advice is to focus on the problems you believe you will get the most out of doing, to maximize understanding per time spent.

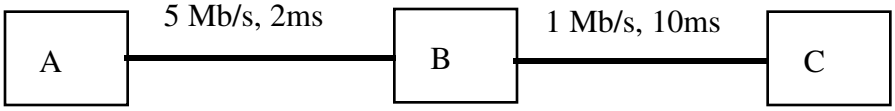
We've set the due date to give you maximum flexibility, but that puts an added burden on you to allocate your time wisely. We advise reading over the problems and letting them simmer, instead of trying to do them all at once.

Since we won't be able to grade the problem sets in time for the final, we will release a solution set immediately after the due date. Please keep a copy to check your own answers.

Questions are divided into "easier" and "harder" sets; these are only relative terms -- you may find some questions in the easier set hard to answer, and others in the harder set easy to answer. It is meant solely as a rough guide. Questions in the easier set are like those that might appear on an exam (several are taken from last year's final). All questions are weighted equally. You are free to discuss this assignment with other students and the TA, but you must each individually write up and submit your own solutions.

Easier Questions

1. Rate/Delay Concept. Suppose three computers (A, B, and C) are interconnected by two links as follows:



a. How long does it take for A to send an 8KB packet to B and for B to fully receive that packet?

b. How long does it take for A to send an 8KB packet to C (via B) and for C to fully receive that packet, assuming that B must receive the entire packet before forwarding it to C (the last bit from A has to arrive at B before B can send the first bit to C -- store and forward)?

c. How long does it take for A to send an 8KB packet to C (via B) and for C to fully receive that packet, assuming the 8KB packet was sent as 16 separate 512 byte chunks, again assuming B is store and forward?

d. Assume now that A sends packets to C continuously (i.e., back to back) at a rate dictated by the link between A and B. Assume B can queue at most 10 packets at any time. How long after A starts blasting packets will B be forced to drop a packet? Beyond that point, how often will B drop packets? If B's queue size is increased to 20 packets, how often will it drop packets (assuming A still sends continuously)?

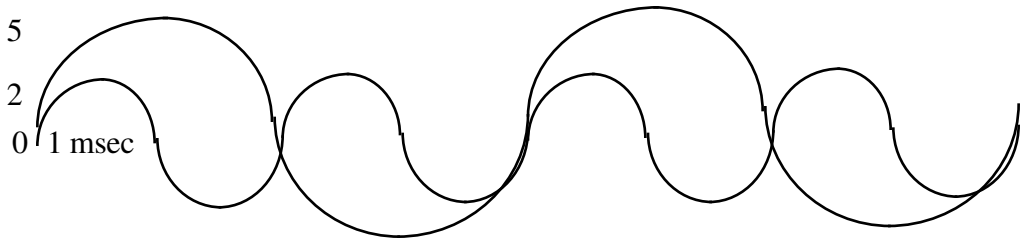
e. Now, instead, assume A sends packets back-to-back until it receives an acknowledgment packet from C, at which time it immediately stops sending. Assume that C generates this acknowledgment as soon as it receives the last bit of the first packet from A. How many packets will A send? Does the queue at B overflow?

2. Derive a formula for the efficiency of a stop and wait protocol, assuming a full duplex link, in terms of: T (the timeout value), p (the probability that the link drops a packet), d (the propagation delay, i.e., the transmission time that would be experienced by a 1 bit packet), r (bandwidth of the link), s (packet size), and h (header size). Hint: it is useful to define intermediate terms, e.g., the round trip time for a packet of size s, etc.

3. Write a pair of RPC stubs to turn your code for homework #1 into a "Microsoft stock quote server". In other words, the client calls "getQuote()", but the actual code to implement the routine runs on a different machine. For extra credit, demonstrate that your code works by running it on two separate machines.

4. The book claims that the initial sequence numbers (ISN) of a TCP connection are chosen "randomly" by each end point. This is not quite true. Find out what TCP actually does to choose ISNs and briefly describe the algorithm. You might look this up in the TCP specification itself, as described in "Request for Comments 793" (RFC-793). To find an on-line copy, try your favorite search engine or go to http://www.ietf.org/, and follow links.

5. The Fourier analysis of an input signal has already been done for you and shown to consist of two sine waves, one with period 2 milliseconds and amplitude 2, and one with period 4 milliseconds and amplitude 5. Suppose the frequency response of the medium scales frequencies at or below 250 Hz by a factor of 1 (they come out as they go in), and frequencies at or above 500 Hz by a factor of 2 (they come out twice as big as how they come in). Reconstruct the input and output signals by constructing a table listing the values of each of the signals at 250 microsecond intervals. Use this table to sketch the input and output signals.



6. Suppose you are transmitting a light wave down a multimode fiber. Assume that each 1 is encoded as a pulse of light that splits into two signals, one which goes directly down the center of the fiber, the other that bounces off the walls of the fiber, thereby taking a longer route. Suppose the short route takes 100 nanoseconds and the long route takes 125 nanoseconds. What is the maximum bit rate we can use on the fiber without causing intersymbol interference? What happens if we triple the length of the fiber link?

7. Consider an encoding where the clock is resynchronized at the beginning of every byte, using 1 start bit (e.g., transition high), 8 data bits (including a parity bit), and 2 stop bits (e.g., transition high to low). Suppose it is possible to have vicious errors that cause the receiver to completely lose synchronization with the sender. Is there a sequence of characters such that the receiver stays out of synch indefinitely (i.e., the receiver detects a valid sequence of characters but the sender is sending a completely different sequence of characters)?

8. Write a program in any convenient language that takes a bit stream (e.g., encoded as ASCII 1's and 0's), and computes an 8-bit CRC on the bit stream, using the polynomial x^8 + x^6 + x + 1 (the bit pattern 101000011).

9. Hugh Hopeful is the President of JTBD (Just to Be Different) Network Company. After consulting marketing, Hugh decides to offer a new framing protocol where the user can specify their own flags to delineate the start and end of frames. One popular framing protocol is HDLC, which uses 01111110 as the start/stop signal; to prevent confusion, HDLC stuffs an extra 0 after ever 5 bits in the user data that are all 1's.

Suppose the user provides an 8 bit flag of the form "aSbc" where a, b, and c are arbitrary bits, and S is an arbitrary 5 bit string. Hugh invents the following bit stuffing rule -- whenever you find the string S in the user data, an extra bit equal to the complement of b is stuffed after it. Notice that Hugh's rule gives exactly the HDLC bit stuffing rule. Can Alyssa P. Hacker find a flag that Hugh's stuffing rule will fail on? Explain.

10. Consider multiple computers attached to an Ethernet hub (a hub is effectively a multiway repeater between a number of Ethernet segments). The maximum allowable distance between a computer and the hub is 100m. Assume the propagation speed of electrical signals on the Ethernet is 1.75 x 10^8 meters/second. Assume also that the hub takes 100 nanoseconds to detect a collision. What is the maximum time between when a node starts transmitting and when it learns that the packet has collided?

11. For each of (i) link state, (ii) traditional distance vector, and (iii) path-based distance vector:

(a) Under what circumstances, if any, is it possible for router A to forward packets for router B through router C, even though routers A and B are neighbors.

(b) Suppose that A and B are neighbors, but A thinks the cost of the link to B is x, and B thinks the cost to the link to A is y. Will this cause any problems? Explain in terms of an example.

(c) Give a topology in which, due to an inconsistency in the information about a single link, many routes are disrupted. Give a topology in which, despite an inconsistency in the information about a single link, no routes are disrupted.

12. Consider an arbitrarily large network in the form of a fully balanced binary tree of L levels. There are 2^k nodes at level k (for k = 0 ... L - 1) and there are 2^L - 1 nodes altogether. Each leaf at the bottom of the tree represents a host and each internal node represents a router. Your task is to devise a scheme for assigning addresses to nodes that is optimal in the sense of minimizing the maximum size of the routing tables across all routers, assuming each entry in the routing tables store only prefixes with unique routes (in other words, the routers are CIDR-based). Indicate what it stored in each router's forwarding table.

13. Consider the following proposal for hierarchical source routing based on the Internet's "loose source routing". With loose source routing, a host specifies a list of routers to visit on the way to the destination; the network is free to use any means of getting between the routers in the loose source route. A hierarchical source routing scheme could be built on top of this, by saying that when a packet is delivered to a router specified in the loose source route list, it can in turn "push on the stack" a loose source route to the next router in the original list. Explain why this might be better or worse than the way hierarchical routing is done in the Internet today. Where appropriate, give concrete examples.

14. Draw a timing diagram illustrating all the packets that are sent between a client and server, when the client requests a 10KB web page, and the previously determined maximum transfer unit without fragmentation is 512 bytes. Include connection setup and teardown, slow start, delayed acks, etc. . Roughly how many round trips does it take? What happens if the third data packet from the server is dropped? How many round trips in this case?

15. Recall that in source routing, the sending host lists each hop of the path explicitly in the packet header. Suppose we want to extend unicast source routing to multicast. Sketch a design of a scheme for doing so. How could you efficiently encode the route of a multicast packet in the packet's header? Sketch the format of the packet header that you would define to implement your scheme. Do you think it would be easy to engineer and deploy such an approach in a real network?

16. One difficulty in the deployment of the Mbone has been incompatibilities between variants of multicast routing protocols. Suppose we want to bridge together two multicast networks, N1 running protocol P1 and N2 running protocol P2. Assume there is a sender S in N1 and a receiver R in N2 and that the two networks are interconnected via two gateways G1 and G2. In other words, there are two possible paths between N1 and N2. Suppose that P2 computes a reverse-path route from R to S via G1 but P1 computes the distribution tree from S to N2 via G2. As a result, packets will not be correctly routed from S to R. Explain how this happens. What might you do to avoid this problem?

17. Assuming nothing is cached, use a timing diagram to illustrate the packets that are sent to load my home page, http://www.cs.washington.edu/homes/tom/index.html? How many round trips does it take? What changes if the name is cached but not the page? What if the page is cached? Assume that the request goes through a name proxy and that "edu", "washington" and "cs" are each in their own zone.

18. Recall that DNS server replicas are updated asynchronously, so that at any point in time, some replicas may have seen an update while others have not. Eventually, each update arrives at each replica.

(a) Explain how a node can modify a DNS record, then try to read back the modified record, and still find the old information. This can occur even if no other node has ever modified the record. Why?

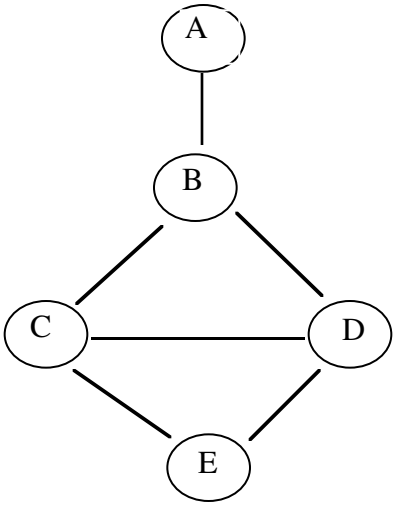
(b) Sketch a solution that enforces "processor order" -- a node always sees its changes to the database in the order that the node made them (e.g., it never sees changes disappear).

(c) Sketch a solution that enforces a "total order" -- every node sees changes to the database in the same order (e.g., if host A modifies record P then modifies Q, if host B reads Q and gets the new value, if it later reads P, it is guaranteed to get P's new value).

19. End-to-end Argument
Almost all network protocols are designed to be robust to failures. For each of the following, explain (i) what would happen if the information was lost, and (ii) how the system would recover, if at all. For example, if one side loses TCP connection state (for example, because it crashes), the TCP connection would be terminated, but the hosts could reestablish communication by making a new connection.

(a) A TCP acknowledgment
(b) A fragment of an IP packet
(c) A distance vector routing table update
(d) A packet carrying link state routing information
(e) An ATM virtual circuit entry
(f) A DNS resource record for an mit.edu host cached on a Washington name server
(g) A DNS resource record for an mit.edu host maintained on its home server at MIT.
(Hint: all DNS servers are replicated.)
(h) A name/address entry in a host's ARP cache
(i) Prune-state in a multicast routing tree
(j) A user's private key in a public key encryption system

20. Routing. In this problem, we consider how distance vector update messages interact with multicast. Consider the following topology:



(a) Assuming that each node initially starts with itself in its routing table (with hop count 0), and at each step a single message is sent, what is the state of the routing tables at each node after the following sequence of seven updates is completed: E->D, D->C, C->B, B->A, A->B, B->D, D->E.

(b) Show the sequence of packets that that will be sent for a multicast packet originating at node C, using reverse path flooding for the network and routing tables from (a). What happens if shortly after C sends out a multicast packet (but before the multicast has reached all nodes), C sends a routing table update to E?

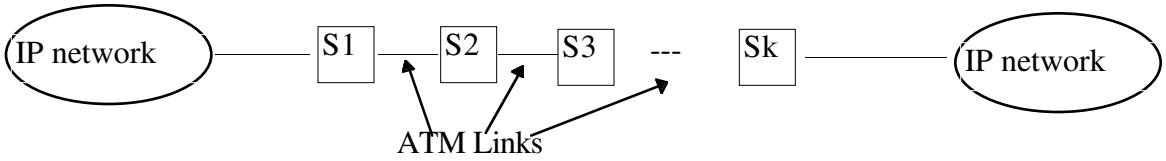
(c) Suppose we send a few more distance vector messages, after the C->E message in part (b): D->B, C->D, B->C, E->C. Show the reverse path broadcast tree for multicasts originating at node B.

21. Security. Which is more powerful, public key or secret ("private") key encryption? One way to answer this question is to ask, can we simulate one with the other? Assume we have three parties, A, B, and C, and either each pair shares a secret key (K[A,B], K[B,C], or K[A,C]), or each node has a public key pair, K[A-priv], K[A-pub], K[B-priv], K[B-pub], K[C-priv], K[C-pub], where the private key is known only to the owner, and the public key is known to everyone.

(a) Can we simulate secret key encryption with public key encryption? If so, show how we would encrypt a message from A to B using public key encryption that provides the same security as the message encrypted in K[A,B] -- in other words, the message could only come from A and be read by B. If this is not possible, explain why not.

(b) Can we simulate public key encryption with private key encryption? If so, show how we can use secret key encryption to provide (i) the same security as a message encrypted only in K[A-priv], and (ii) the same security as a message encrypted only in K[A-pub]. If either of these is not possible, explain why not. You can assume C is trusted by everyone and has a secure disk; for example, it can store messages securely for later retrieval.

22. IP switching. Suppose we are using ATM links to carry packets between Internet routers. Here's the scenario. We have **k** routers connected by k-1 ATM links; each link runs at **b** bits per second, the IP packet size is **p** bytes, the ATM cell size is 53 bytes consisting of 5 bytes of header and 48 bytes of useful data, and the propagation delay on each link is **d** seconds. Assume that the average queue length at each router is **q** and that router table lookup takes negligible time. Sample values are k = 4, b = 155 Mb/s, p = 2400 bytes (including the IP header), d = 1 millisecond (0.001 second), and q = 2.



(a) S1 - Sk are IP routers but the links carry ATM cells. When an IP packet arrives that is routed over an ATM link, the packet is fragmented by the router into ATM cells which are then sent over the link. When the cells arrive at the next router, they are reassembled into an IP packet, and the router looks up a route in its IP forwarding table to find the next link. If the packet is routed over another ATM link, it is refragmented back into cells, and transmitted to the next router. Write an expression in terms of **k**, **b**, **p**, **d**, and **q** for the average time for the packet to traverse the k routers, from the time it completely arrives at S1 until it is ready to be sent out the link from Sk into the rest of the Internet. Calculate the result for the sample values above.

(b) A number of router vendors have recently added support, called IP switching, to optimize for this case where a number of ATM links are used in sequence. The routers can "bind" an IP flow (sequence of packets) to a virtual circuit that spans multiple routers, in this case, from S1 to Sk. The IP flow is identified by the (IP source address, IP destination address) of the IP header. Once the binding is complete, when a packet arrives at S1, S1 notices that a virtual circuit (such as VC7) has already been set up for that flow, and S1 fragments the packet into cells sent on VC7. When they arrive at S2, the router pretends it is an ATM switch, and forwards the cells as they arrive to S3, and so forth, until they arrive at Sk, where they are reassembled into an IP packet. Assuming the virtual circuit has already been set up, write an expression as in part (a) for the time for a packet to traverse the k routers. Calculate the result for the sample values above.

(c) Although binding IP flows to ATM connections reduces forwarding delay, the binding process initially incurs a delay at each router to set up the connection. Accordingly, it is not advisable to set up a virtual circuit for every packet that shows up at the edge of an IP. Suppose it takes **T** seconds to set up a virtual circuit at a router/IP switch, write an expression for how big a packet must be for it to make sense to set up a virtual circuit just for that packet. Suppose T = 1 millisecond; would it make sense for the sample values given above? (Of course, the overhead of setting up the virtual circuit can usually be amortized across multiple packets between the same hosts.)

23. TCP. Recently, a number of wireless networks for carrying digital Internet traffic have been deployed; one example is the Metricom packet radio system which can be used for wireless communication on the UW campus. Wireless communication links are typically much more likely to suffer bit errors than traditional "wired" networks, with drop rates approaching 1 in 4 packets. These networks are also usually just for the final "hop"; Internet packets traverse some number of wired links and routers before being sent over the wireless link, which delivers packets directly to the receiving host (for example, a portable PC with a cellular phone and modem).

(a) When these wireless networks were first deployed, observed TCP bandwidth over the wireless link was usually an order of magnitude worse than the peak bandwidth the hardware could deliver. Explain why.

(b) One proposed solution to this problem is to cache packets at the upstream router just prior to being sent over a wireless link. If the packet is garbled, the packet can be resent without having to wait for a retransmission from the source host. Of course, this solution is easier to deploy if it doesn't require changing the protocol stack at either the sender or the receiver. Consider the case where the packets are being sent to a wireless host and all the traffic is TCP. Can an intelligent router detect when retransmissions are needed without modifying the receiver? If so, how? If not, why not? Hint: the router can observe TCP acknowledgments.

More Difficult Questions

24. Derive a formula for the efficiency of a sliding window protocol, with window size X packets, assuming that the packet loss rate (p) is 0, in terms of: T (the timeout value), d (the propagation delay), r (bandwidth of the link), s (packet size), and h (header size).

25. Derive a formula for the efficiency of a go-back N protocol assuming at most one packet is dropped per window (X < 1/p).

26. Derive a formula for the efficiency of a selective acknowlegment sliding window protocol, assuming the window size is arbitrarily large.

27. Derive a formula for the efficiency of a TCP connection, once it has reached steady state, as a function of its packet loss rate (recall that TCP interprets a packet loss as a signal that there is congestion, causing it to reduce its rate by half). Assume at most one packet is dropped in each congestion period (at the tip of the sawtooth), and fast retransmit is used to recover. (Hint: the result should be proportional to the packet size, and inversely proportional to the round trip time and the square root of the packet drop rate). Since the exact solution is complex, it is reasonable to ignore less important terms, but be clear what assumptions you are making.

28. One difficulty with using round-trip time estimation to control retransmission timers is accounting for the variability of the actual round trip times. Suppose we use an exponentially weighted moving average to keep track of the estimated round trip time. Let r(n) be the n'th round trip time measurement, and y(n) be the estimated value after n measurements, then we compute: y(n) = a * y(n - 1) + (1 - a) * r(n). Next suppose that we set the timeout to be B* y(n).

If the round trip time is in steady state, with the estimate y(n) equal to the real round trip time z, and the round trip time jumps to a new value s >> z, the system will incorrectly timeout on each packet while the estimate gradually converges to the new value. How long will this convergence take -- how many iterations will it take before the estimate converges close enough to the new value to avoid an unnecessary timeout (B * y(k) > z). Show that when the number of iterations is >= log (1 - 1/B) / log(a), the unnecessary timeouts cease.

29. In this problem, you need to write some clock recovery code for receiving a 4-5 encoded bit stream. Assume the preamble has been received and the receiver is in sync except for possible clock drift. Thus, the receiver is sampling according to its current clock, but because of clock drift it may be a little off. Remember that in 4-5 coding you ar guaranteed to get at least one transition in every 5 consecutive bits; you may get up to 5 transitions, however. (Hint: you may want to receive a group of 5 bits, keep track of how many transitions you sample, and use this to adjust the receiver clock for the next 5 bits. Does it help to oversample -- e.g., take 10 samples to determine if you are missing some transitions?) You can assume there is no noise. Extra credit if you have a sensible strategy to deal with noise.

30. Suppose that each bit in a frame has an independent 50-50 probability of being a 1 or a 0. This might happen, for example, if the data is first encrypted and/or compressed, before being sent. The HDLC bit stuffing rule is to add a 0 after ever 5 consecutive 1's in the user data. What is the average overhead of HDLC? (Hint: The obvious answers like 1/32 or 1/64 are wrong.)

31. Unlike unicast addresses, which are semi-permanently assigned to hosts, multicast addresses are often allocated dynamically. That is, a multicast address must somehow choose an address for its traffice -- how does it do this?

(a) One approach would be to create a central authority that would maintain a pool of addresses and "lease them out" when a group of hosts wants to start a multiway communication. Cite two fundamental problems with this centralized approach.

(b) An alternative approach is based on random address allocation. One problem with this scheme is the potential for address collision. Suppose the addresses are allocated randomly from among Z total multicast addresses. If we assume that hosts do not share information about which addresses have already been selected, what is the probability that a collision occurs given N allocations (by N independent hosts)? If Z is 64K, for what value of N does the probability of a collision exceed 1%? Suppose instead that information could be shared on a limited basis. In this case, if we assume that N allocations are carried out each second and each host learns of allocations after one second, and if we assume each address is held for an hour, what is the probability of a collision as a function of N?