CS268 Networking	**Spring 1996**

Midterm 1 : Date February 22

Time Limit: 2 hours *Instructors and TA: Kevin Fall, Michael Luby, and Elan Amir*

Name: ______________________________

Basic Instructions

For the following questions, give a short, concise answer that is at most one or two paragraphs long. Be as concrete as possible, and substantiate your arguments with specific examples wherever possible. Much more credit will be given for a good justification of a good answer than for just a good answer. Each of the seven questions is worth an equal number of points.

General Networking

(1) (a) What is the difference between a bridge and a router?

(b) What is the end-to-end argument? Give a specific example in network system design where this principle is used.

(c) What is a (network-layer) gateway and why does the Internet need gateways?

(d) Explain the differences between connection oriented and connection-less protocols.

ALOHA, Ethernet and MAC Layer

(2) (a) What is the basic idea behind "spread ALOHA"?

(b) Explain why the length of a single Ethernet cable must be limited.

(c) Assuming ideal conditions (e.g., no signal deterioration, assume no headers), can an Ethernet frame be 10 bytes long? If not, explain why. If so, explain what constraints does this choice impose on the system.

(d) ARP is often cited as a security weakness. Explain why.

Wide Area Networks

(3) **(a)** The 'A' in ATM stands for "Asynchronous". What does this mean? What are the fundamental differences between ATM and STM, and what are the implications of these differences. (Recall that "STM" stands for "Synchronous Transfer Mode".)

(b) What is the difference between FPS (fast packet switching) and pure packet switching? What are the relative tradeoffs?

(c) Wide-area multicasting is significantly more challenging than wide-area unicasting. Why is this so?

Hamming Codes

Notation: In an (n, k) Hamming Code, each code word is n bits long, of which k bits are message bits and the remaining $n - k$ bits are redundant bits.

(4) (a) What is the Hamming distance between codewords in a $(7, 4)$ Hamming code?

(b) Up to how many errors can a $(7, 4)$ Hamming code reliably detect?

(c) Up to how many errors can a $(7, 4)$ Hamming code reliably correct?

Information Theory

(5) Consider the following method for generating a code for a random variable X which takes on m values $1, 2, \ldots, m$ with probabilities $p_1, p_2, \ldots, p_m$. Assume that the probabilities are ordered so that $p_1 \geq p_2 \geq \cdots \geq p_m$. Define $F_1 = 0$, and for all $i \in \{2, \ldots, m\}$, define

$$F_i = \sum_{k=1}^{i-1} p_k,$$

to be the sum of the probabilities of all symbols less than i. Then the codeword for i is the number $F_i \in [0, 1]$ rounded off to ℓ_i bits, where $\ell_i = \lceil \log \frac{1}{p_i} \rceil$.

(a) Show that the code constructed by this process is prefix-free and the average length, L, satisfies

$$H(X) \leq L < H(X) + 1$$

(b) Construct the code for the probability distribution (0.5, 0.25, 0.125, 0.125)

Ethernet Protocols

(6) Consider the following "slotted" version of Ethernet. At each time slot, each of the processors connected to the Ethernet may elect to transmit information. This is controlled by a variable p_i local to processor i. When a message arrives to processor i and processor i has no other messages at that point in time, then p_i is set to 1. By the time the previous slot terminates and the current slot begins, processor i knows one of the following is true, and behaves as described. (All other processors behave similarly.)

- **(0)** *No processor tried to transmit in the previous slot.* If processor i has a message to transmit then it resets $p_i \leftarrow 1$ and tries to transmit the first part of the message in the current slot.
- **(1)** *Exactly one processor tried to transmit in the previous slot (and was successful without collision).* Processor i tries to transmit only if it is the processor that successfully transmitted in the previous slot. If processor i still has part of the message to be sent, then this is transmitted in the current slot. If the last portion of the message was transmitted by processor i in the previous slot then nothing is transmitted in the current slot.
- **($>$ 1)** *More than one processor tried to transmit in the previous slot (thus a collision was detected).* If processor i has a message to transmit then reset $p_i \leftarrow p_i/2$. Processor i tries to transmit in the current slot with probability p_i.

In all of the following analyses, suppose Q processors have a message to transmit and no processor tried to transmit in the previous slot, and suppose no new messages arrive subsequently. Express all answers in terms of Q. *Hint: Do not try to compute the answer exactly, but instead make some approximations. All answers should be correct to within a small constant multiplicative factor (independent of Q), and justification should be given.*

(a) What is the expected number of slots until there is a slot where less than two processors attempt to transmit?

(b) For all $i = 1, \ldots, Q - 1$, what is the expected number of slots between the time the i^{th} message successfully completes transmission and the time the $(i+1)^{rst}$ message successfully starts transmission.

(c) Suppose the protocol is modified so that if no transmission occurred in the previous time slot (case (0)) then, instead of resetting $p_i \leftarrow 1$, p_i is reset to the minimum of 1 and $2p_i$. This means that processor i has some "memory" of past collisions. Argue that the average number of slots between the time one message successfully completes transmission and the next message successfully starts transmission is a constant independent of Q.

(d) In the original protocol, each processor that has a message to send at the beginning of a free time slot has an equal chance of being the next to successfully transmit its message, even if new messages arrive. What kind of "fairness" properties does the modified protocol have?

Sorting Protocols

(7) Recall that $a_1, \ldots, a_n$ is a *bitonic sequence* if it increases and possibly decreases, i.e., if there is some $i \in \{1, \ldots, n\}$ such that $a_1 \leq a_2 \leq \cdots \leq a_i \geq a_{i+1} \geq \cdots \geq a_n$. We say $a_1, \ldots, a_n$ is a *sorted sequence* if it is an increasing sequence, i.e., if $a_1 \leq a_2 \leq \cdots \leq a_n$. An n-*bitonic sorter* accepts as input a bitonic sequence $a_1, \ldots, a_n$ and produces a sorted sequence $b_1, \ldots, b_n$ such that $\{b_1, \ldots, b_n\}$ is the same as $\{a_1, \ldots, a_n\}$ as a multiset. An n-*sorter* accepts as input an arbitrary sequence $a_1, \ldots, a_n$ and produces a sorted sequence $b_1, \ldots, b_n$ such that $\{b_1, \ldots, b_n\}$ is the same as $\{a_1, \ldots, a_n\}$ as a multiset.

Let $n = 2^p$ for some positive integer p. Describe how to construct an n-sorter using 2^i-bitonic sorters as basic building blocks (where $i \in \{1, \ldots, p\}$). Your circuit should have at most p levels of bitonic sorters. Think inductively, and remember, a picture is worth a thousand words in your description.