

Exam 1 (Midterm), Spring 2002  
Computer Science 516  
Computer Communication Networks  
University at Albany

Prof. William A. Maniatty

March 25, 2002

## 1 Some Hints

Your professor suggests the following preparation strategies:

- This exam is open book and open notes (your own books and notes that is!). Calculators are permitted, networked devices are not.
- Write neat clean answers, since if the grader cannot understand you on the real exam, it will go badly for you.
- Show your work, if you are guessing the grader will not give much credit (even if you get lucky and guess right).
- Define your notation (you can use tables and/or diagrams).
- Set up the solution symbolically and simplify before plugging numbers in, it is easier to follow for the grader (and will get you most of the points).
- You can solve problems out of order, but keep the work for each problem in one place, and mark it clearly.

## 2 The Problems

### 1. Protocol Layering and Design (25 %):

- (a) Briefly describe the services that the data link layer provides.

The data link layer provides point to point communication across dedicated channels or broadcast media. Signals processed by the physical layer are aggregated into information bearing *frames* by the data link layer. Frame separators are maintained by the data link layer as well as providing some indication that the link is currently idle (since some signal is always received, an idle signal is specified). Finally for broadcast media, a mechanism is provided so that the intended recipient receives the messages while other nodes ignore it (via addressing) and scheduling of whose turn it is to transmit is handled at this layer. Data link layers are the lowest layers implementable in software but are typically hardware implemented for speed.

- (b) (5 %) Compare and contrast the goals (*not* the mechanisms) of ARQ and flow control (5 %).

ARQ focuses on providing reliable data communication over an unreliable channel. ARQ ensures that packets are not delayed beyond a timeout threshold, that the packets arrive uncorrupted and that packets are received in the order in which they were sent. Flow control focuses on matching sender, receiver, and link bandwidth capacity and adapting to congestion.

- (c) What services does a *reliable* protocol provide (5 %)?

A reliable protocol ensures the following properties:

- i. It does not drop packets
- ii. Packets are received in the order in which they were sent.
- iii. Packets are not corrupted.

If one of these properties cannot be ensured, an error will be reported to the sender. Typically a timeout is needed to ensure packet loss does not occur.

- (d) For end-to-end connections, if frame sizes exceed the Maximum Transfer Unit (MTU), some node along the way will need to fragment the packet. The TCP protocol suite has an end-to-end protocol, the path MTU discovery (PMTU) protocol for estimating the MTU.

- i. At what layer is the MTU establishment protocol implemented (5 %)?

PMTU should be implemented at the transport layer, since it relies on the end-to-end connectivity provided by the network layer.

- ii. Should PMTU be implemented as a data protocol or as a control protocol? Justify your answer (5 %).

This is a control protocol, since it reports on and permits regulating metadata (in particular packet size), and does not influence the user data.

2. Network Systems programming (10 %): Assume a BSD style sockets programming environment.

- (a) (10 %) How do datagram and streaming sockets differ?

Both datagram and streaming sockets send and receive packets of data. Streaming sockets give the *appearance* (via the interface) of transmitting data one byte at a time. Datagram sockets expose the sending and receipt of packets to the receiver.

- (b) (5 %) What functionality does the listen system call provide?

The listen system call waits for a connection attempt on a socket. Listen is invoked on the server side, and takes two parameters, the socket that is waiting for a connection and the queue length (backlog) of connections allowed for that socket.

3. The Physical Layer and Information Theory (25 %):

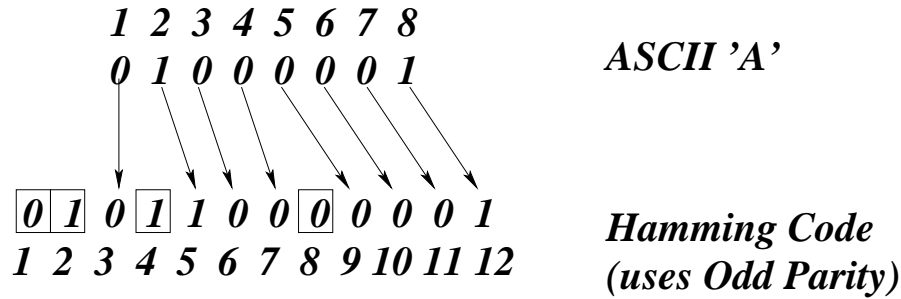
- (a) (10 %) Modern telephone systems use pulse code modulation (PCM). During transmission of a signal, distortion may occur, recall how pulse code modulation can recover from errors. Suppose a device using PCM has voltages over  $+0.5$  and  $-0.5$  volts with 128 signal levels. What is the maximum level of signal distortion that can be corrected?

The signal levels are assumed to be uniformly spaced over the signal range. The signals are spread out over the range  $[-0.5V, +0.5V]$ , giving a  $0.5 - (-0.5) = 1$  volt signal range. The number of levels is  $2^7 - 1 = 127$ , so each signal level is  $\frac{1}{127}$  volts apart. Suppose the received voltage falls between two of the signal levels. The correct value is assumed to be the nearest signal level. This means it cannot be more than  $1/2$  the distance to the nearest incorrect signal level or recovery will fail. Thus the maximum recoverable error is thus  $\frac{1}{127}$  volts  $\times \frac{1}{2} = \frac{1}{254}$  volts = 0.0039volts.

- (b) (10 %) Suppose your grade for this course is being transmitted across noisy channel (the horror!). Suppose the grade, an 'A', is encoded in ASCII as  $0x41$  (in decimal that would be 65).

- i. Derive a Hamming code using odd parity (so the parity ensures that an odd number of bits are 1) and that the bits are numbered starting at 1 with check bits are at the power of two positions (as per the lecture).

The Hamming code is as shown in Figure 1.



**Parity Bits not detecting errors**

Figure 1: Hamming Code from Problem 3(b)i

- ii. Suppose that the noise corrupts the Hamming Code transmission in problem 3(b)i, so that the data bits are corrupted, and the data sent is an ASCII 'E', (hexadecimal  $0x45$ ). The metadata (the parity bits) are not corrupted. Show the Hamming code, can the the error be detected and corrected (5 %)? Why or why not?

The Hamming code is as shown in Figure 2. Since there is only one bit error (bit 6 in the unencoded data, and bit 10 in the encoded data is in error), the Hamming code detects this error and correctly reports the position of the corrupted bit. Thus this error is correctable.

- iii. Suppose that the noise corrupts the Hamming Code transmission in problem 3(b)i previous step, so that the data bits are corrupted, and the data sent is an ASCII 'B', (hexadecimal  $0x42$ ). The metadata (the parity bits) are not corrupted. Show the Hamming code (5 %), can the the error be detected and corrected (5 %)? Why or why not?

The Hamming code is as shown in Figure 3. This time there are two bit errors (bits 7,8 in the unencoded data, and bits 11,12 in the encoded data are in error), the Hamming code detects this error, but it incorrectly reports the position of the corrupted bits (because it does not have enough metadata to report two different bit errors). Thus this error is detected, but not correctable. If

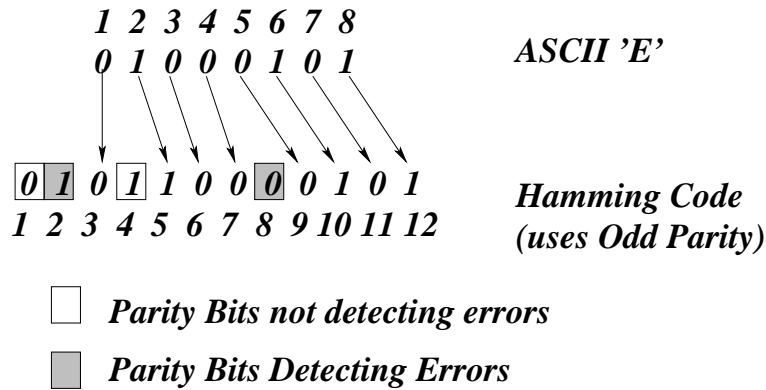


Figure 2: Hamming Code from Problem 3(b)ii

we applied traditional Hamming Code style correction, we would change bit 7 in the Hamming code, instead of fixing bits 11 and 12.

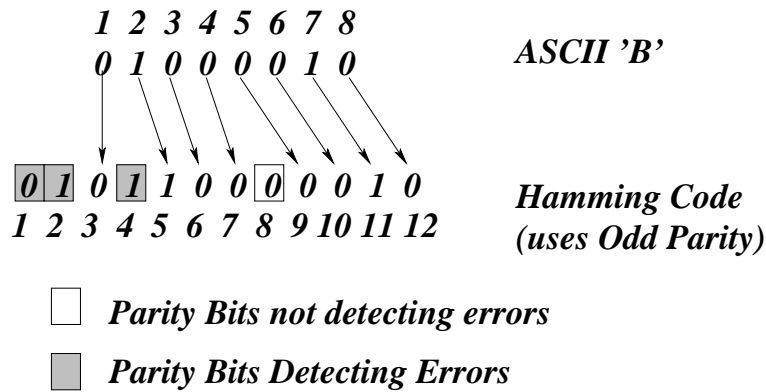


Figure 3: Hamming Code from Problem 3(b)iii

- (c) (5 %) Consider the CRC which has the generating polynomial:  $G(x) = x^5 + x + 1$ . Give the CRC for the message  $10101011_2$ .

For notational convenience, we will use the one's complement notation for representing values and all values will be positive. First, we develop a representation of the generating polynomial using the coefficients to generate the binary value 100011. We then take the message and shift it left the distance of the highest order term in the generator polynomial, yielding 1010101100000. We now carry out the "long division" using the modulo 2 arithmetic, as follows, with the

remainder being the CRC:

```

          10100100
          -----
100011 | 1010101100000
          100011|||||
          -----VV||||
          100111|||||
          100011|||||
          -----VVV|
          100000||
          100011||
          -----VV
          01100 <- CRC
  
```

For additional details on the procedure, please see the text book by Leon-Garcia and Widjaja [ ] and the lecture notes.

4. ARQ and Performance (25 %) Consider a packet switched network with a sender and a receiver. Suppose that the network has a mean propagation delay of 0.01 seconds, and a mean bandwidth of 12 Mbps. Due to heavy congestion, the packet loss rate is 1% on average. Suppose that the mean information frame size is 1500 Bytes, and the packet header and trailer meta data is 80 Bytes (constant). Suppose that ack and nack frames are 80 Bytes (just headers, they contain no data).
- (a) What is the efficiency of a stop-and-wait protocol in the presence of errors for this scenario and how many information frames could have been sent while waiting for the acknowledgment (10 %).

Borrowing the notation from the lectures notes and the text book, the notation looks like (with given values specified).

Symbol	Meaning
$\eta$	Efficiency of the Protocol with errors
$\eta_0$	Efficiency of the Protocol Without Errors
$n_a = 80$	ACK frame size
$n_f = 1500\text{Bytes}$	I-Frame size
$n_o = 80\text{Bytes}$	frame overhead
$p_f = 0.01$	Per packet error probability
$R = 12 \times \frac{10^6 \text{bits}}{\text{sec}}$	Link Bandwidth (symmetric)
$R_0^{EFF}$	Effective transmission rate
$t_0$	time to send a frame and get an ACK
$t_a$	ACK transmission time
$t_f$	I-Frame transmission time
$t_{prop} = 0.01\text{sec}$	Propagation delay
$t_{proc}$	Time to process header or ACK

Since  $t_{proc}$  was not specified, and nobody asked during the exam, let's assume it is negligible (which is not unreason-

able given today's fast hardware, thus we assume  $t_{proc} = 0$ ). What we want to do is solve for  $\eta$ , but we will need to estimate  $\eta_0$  first, which is also unknown. Following our equations given in the lecture notes (I used one of the simpler forms showing in the derivation, not the final stage), we substitute, normalize all data sizes to bits and get:

$$\begin{aligned} \eta_0 &= \frac{n_f - n_0}{2R(t_{proc} + t_{prop}) + n_f + n_a} \\ &= \frac{(1500 - 80)\text{Bytes} \times \frac{8\text{bits}}{\text{Byte}}}{2 \times 12 \times 10^6 \frac{\text{bits}}{\text{sec}} \times ((0 + 0.01)\text{sec}) + (1500\text{Bytes} + 80\text{Bytes}) \times \frac{8\text{bits}}{\text{Byte}}} \\ &= 0.045 \\ \eta &= (1 - p_f)\eta_0 = (1 - 0.01) \times 0.045 = 0.044 \end{aligned}$$

which is quite low, the impact of lost packets is slight as  $p_f$  is small.

- (b) Suppose go-back-N ARQ is used with a sender window size of 64 and we are considering increasing the window size to 256. What is the efficiency of the protocol in the presence of errors for each window size, which window size should we choose (10 %)?

The approximation given in the book and lecture notes is acceptable here, but it is not quite perfect (it appears to reward small window sizes regardless of the delay bandwidth product, acknowledgements appear to be treated as negligible). Using the notation from 4a, and substituting the known values we get:

$$\begin{aligned} \eta &= (1 - p_f) \frac{1 - \frac{n_0}{n_f}}{1 + (W_S - 1)p_f} \\ \eta &= (1 - 0.01) \frac{1 - \frac{80\text{Bytes}}{1500\text{Bytes}}}{1 + (W_S - 1)0.01} \end{aligned}$$

Solving for when  $W_S = 64$  we get  $\eta = 0.57$  and for when  $W_S = 256$  we get  $\eta = 0.264$ , much better! However, let's check the delay bandwidth product (normalized to frames) to see if our window sizes are too small.

$$R \times 2t_{prop} = 12 \times 10^6 \frac{\text{bits}}{\text{sec}} \times 2 \times 0.01 \text{sec} \times \frac{1\text{Frame}}{1500\text{Bytes} \times 8 \frac{\text{bits}}{\text{Byte}}} = 20.0$$

So the window is big enough to keep the pipeline filled, yet still allow recovery from errors for both sizes, making  $W_S = 64$  the better choice.

- (c) Suppose that the window size selected in problem 4b is used, but we are interested in changing the protocol to use selective-repeat ARQ.

What would the efficiency of the protocol in the presence of errors be (5 %)?

We can rely on information from our solutions to Problems 4a and 4b. Using the formula from the lecture notes and using our knowledge that the window size is “large enough”, the approximation is *insensitive to window size, delay and bandwidth*.

$$\eta = (1 - p_f)(1 - \frac{n_o}{n_f}) = (1 - 0.01)(1 - \frac{80}{1500}) = 0.937$$

5. Multiplexing and Telephony (15 %) Consider a concentrator, with an offered load is 50 Erlangs.

(a) What is the minimum number of output trunks required to handle that load with a blocking probability of 1%? Show your work (5 %).

The Erlang probability for blocking is:

$$P_b = \frac{\frac{a^c}{c!}}{\sum_{k=0}^c \frac{a^k}{k!}}$$

Where  $a$  is the offered load, and  $c$  is the number of servers. Leon-Garcia and Widjaja suggest that 64 trunks are needed to get  $P_b = 1\%$ .

(b) Suppose that the concentrator’s load is expected to increase to 60 Erlangs over the next 5 years, after which the cabling will be upgraded with an anticipated 10 fold increase in bandwidth. In the short term, assume that the system can handle this year’s load, but will be in difficulties starting next year. To handle this load we need to act now, and we have two choices. Satellite bandwidth is available for the projected 4 years at  $\$5 \times 10^7$  per year to reroute the projected surplus. Alternatively we can add additional trunks at  $\$10^8$  dollars to add cable with an extra  $\$5 \times 10^6$  per added trunk (each trunk can handle 1 Erlang of load). Which solution should we choose (10 %)?

As per Leon-Garcia and Widjaja, you would need 75 trunks to meet the bandwidth requirements. Thus to handle the increased load without increasing blocking probability, you will need  $11 = 75 - 64$  trunks. The cost of adding these trunks is

$$\begin{aligned} \text{Cost of 11 trunks} &= \text{Fixed Deployment Cost} + 11\text{trunks} \times \text{Cost Per Trunk} \\ &= \$10^8 + 11\text{trunks} \times 5 \times \frac{\$10^6}{\text{trunk}} = \$155 \times 10^6 \end{aligned}$$

The satellite cost is

$$\begin{aligned} \text{Cost of Satellite Rental} &= \text{Number of Years} \times \text{Cost per Year} \\ &= 4\text{years} \times \frac{\$5 \times 10^7}{1\text{year}} = 200000 \end{aligned}$$

And we see that adding trunks is the cost effective solution (assuming accurate load predictions were made).