Name: SOLUTIONS

Department of Electrical Engineering and Computer Science

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

6.02 Fall 2010

Quiz II

November 2, 2010

"×" your section	Section	Time	Room	Recitation Instructor
☐	1	10-11	36-112	Tania Khanna
☐	2	11-12	36-112	Tania Khanna
☐	3	12-1	36-112	George Verghese
☐	4	1-2	36-112	George Verghese
☐	5	2-3	26-168	Alexandre Megretski
☐	6	3-4	26-168	Alexandre Megretski

There are **17 questions** (some with multiple parts) and **12 pages** in this quiz booklet. Answer each question according to the instructions given. You have **120 minutes** to answer the questions.

If you find a question ambiguous, please be sure to write down any assumptions you make. **Please be neat and legible.** If we can't understand your answer, we can't give you credit! *And please show your work for partial credit.*

Use the empty sides of this booklet if you need scratch space. You may also use them for answers, although you shouldn't need to. *If you use the blank sides for answers, make sure to say so!*

Please write your name CLEARLY in the space at the top of this page. NOW, please!

One two-sided "crib sheet" allowed. No other notes, books, calculators, computers, cell phones, PDAs, information appliances, carrier pigeons carrying messages, etc.!

Do not write in the boxes below

1-2 (x/18)	3-5 (x/14)	6-9 (x/18)	10-12 (x/15)	13-14 (x/15)	15-17 (x/20)	Total (x/100)

I Linear Block Codes

1. [6 points]: For each of the three codes below, circle whether the code is a linear block code over $\mathbb{F}_2$ or not. Also fill in the rate of the code.

A. $\{111, 100, 001, 010\}$. Linear / Not linear. Not linear. Code rate = 2/3.

B. $\{00000, 01111, 10100, 11011\}$. Linear / Not linear. Linear. Code rate = 2/5.

C. $\{00000\}$. Linear / Not linear. Linear. Code rate = 0.

Solution: Part A is "Not linear" because the 000 codeword is missing—because the sum of any two codewords must be a codeword for a linear code, the absence of 000 makes the code nonlinear by simple inspection. The others are linear because the sum of any two codewords is indeed a codeword.

The code rate of a block code with 2^k codewords is k/n, where n is the number of bits in each codeword. That's because we can identify each codeword uniquely with k bits, if we have 2^k total codewords. Although most students answered this question correctly, some were confused by the definition of the code rate and answered (wrongly) that it was $1/2^k$ (i.e., the reciprocal of the number of code words in the code). Some students also wrongly answered that the code rate for Part (C) was non-zero (e.g., 1/5). The code rate is 0: if a code has only one element, *no useful information* can get sent because the receiver already knows *exactly* what it's getting!

2. [12 points]: Recall that a block code takes a set of k-bit messages and produces n-bit codewords, with a minimum Hamming distance of d between any two codewords. For each (n, k, d) combination below, state whether a linear block code with those parameters exists or not. *Please provide a brief explanation for each case*: **if such a code exists, give an example; if not, you may rely on a suitable necessary condition.**

A. $(31, 26, 3)$: **Yes** / **No** (circle one)
Solution: Yes. A Hamming code with these parameters exists, as can be easily verified.

B. $(32, 27, 3)$: **Yes** / **No** (circle one) Solution: No. $2^{32-27} = 32 < 32 + 1 = 33$, so such a code is impossible.

C. $(43, 42, 2)$: **Yes** / **No** (circle one) Solution: Yes. The parity code constructed over all 42-bit messages has these parameters.

D. $(27, 18, 3)$: **Yes** / **No** (circle one) Solution: Yes. The rectangular code with $r = 6$ rows and $c = 3$ columns (or vice versa). $n = rc + r + c$ and $k = rc$. The code can correct all single bit errors, as seen in Lab 4.

E. $(11, 5, 5)$: **Yes** / **No** (circle one)
Solution: No. $d = 5$ means **all patterns of up to two** bit errors can be corrected, but $2^{11-5} = 64 < \binom{11}{2} + 11 + 1 = 67$.

Solution: Some students were confused about the difference between the necessary condition for a code to exist and the existence of a code just because a necessary condition holds. Specifically: *if* a linear block code has the ability to correct all single bit errors, *then* its parameters, n and k must satisfy the inequality $2^{n-k} \geq n+1$. This result was proved in the lecture notes. Moreover, we showed that a Hamming code exists for specific values of n of the form $2^m - 1$. However, based on what we studied, we cannot conclude that *if* the relation $2^{n-k} \geq n+1$ holds for a code, then that particular code can correct all single bit errors. So, for example, for Part (D), one needed to show an example of a code with the specified parameters; likewise for Part (A). Part (E) was the trickiest of the problems because it asks about a code capable of correcting *up to two* bit errors, but is a direct application of a problem in the lecture notes and the review problem set. For Part (C), $d = 2$, so the inequality above does not apply. One just needs to observe that there always exists an $(n, n-1, 2)$ code—adding a parity bit to each bit string!

II "Pairwise" Codes

Pairwise Communications has developed a linear block code over $\mathbb{F}_2$ with three data and three parity bits:

$$\begin{aligned} P_1 &= D_1 + D_2 \quad \text{(Each } D_i \text{ is a data bit; each } P_i \text{ is a parity bit.)} \\ P_2 &= D_2 + D_3 \\ P_3 &= D_3 + D_1 \end{aligned}$$

3. [4 points]: Fill in the values of the following three attributes of this code:

1. Code rate = $3/6 = 1/2$.

2. Number of 1s in a minimum-weight non-zero codeword = 3. (Explain your answer.)
   Note: The *weight* of a codeword is the number of 1s in it.

Solution: Assuming without loss of generality that a codeword has the form $D_1D_2D_3P_1P_2P_3$, the seven non-zero codewords for the code are 001011, 010110, 011101, 100101, 101110, 110011, and 111000. It is easy to see that the minimum weight of these codewords is 3.

3. Minimum Hamming distance of code = 3.

Solution: That's because the desired quantity is always equal to the weight of the minimum-weight non-zero codeword for any linear code.

4. [6 points]: The receiver computes three syndrome bits from the (possibly corrupted) received data and parity bits: $E_1 = D_1 + D_2 + P_1, E_2 = D_2 + D_3 + P_2$, and $E_3 = D_3 + D_1 + P_3$. The receiver performs maximum likelihood decoding using the syndrome bits. For the combinations of syndrome bits in the table below, state what the maximum-likelihood decoder believes has occured: no errors, a single error in a specific bit (state which one), or multiple errors.

$E_3E_2E_1$	**Error pattern** ["No errors" / "Error in bit ..." (specify the bit) / "Multiple errors"]
0 0 0	No errors.
0 1 0	Error in P_2.
1 0 1	Error in D_1.
1 1 1	Multiple errors.

5. [4 points]: Alyssa P. Hacker extends Pairwise's code by adding an *overall parity bit*. That is, she computes $P_4 = \sum_{i=1}^{3}(D_i + P_i)$, and appends P_4 to each original codeword to produce the new set of codewords. What improvement in error correction or detection capabilities, if any, does Alyssa's extended code show over Pairwise's original code? **(Explain your answer in the space below.)**

Solution: Adding a parity bit to each codeword increases the minimum Hamming distance from 3 to 4. That does **not** change the error correction capability in the worst case, because there are patterns of 2-bit errors that cannot be corrected, but it increases the error detection capability from all 2-bit errors to all 3-bit errors. Several students were tripped up on this question, which applies a property we did study (in a different context): adding a parity bit to every codeword in a code whose d is odd makes it a code with Hamming distance $d+1$, an even number, and makes it possible to detect one more error than before. But the worst-case error correcting capability remains unchanged.

III Convolutional Coding

Consider a convolutional code whose parity equations are

$$\begin{aligned} p_0[n] &= x[n] + x[n-1] + x[n-3] \\ p_1[n] &= x[n] + x[n-1] + x[n-2] \\ p_2[n] &= x[n] + x[n-2] + x[n-3] \end{aligned}$$

6. [3 points]: What is the rate of this code? How many states are in the state machine representation of this code as discussed in 6.02?

Code rate = 1/3.

Number of states in the state machine representation = $2^3 = 8$.

7. [7 points]: Suppose the decoder reaches the state "110" during the forward pass of the Viterbi algorithm with this convolutional code.

A. How many predecessor states (i.e., immediately preceding states) does state "110" have?
Solution: 2.

B. What are the bit-sequence representations of the predecessor states of state "110"?
Solution: "100" and "101". Note: we assumed (as in the rest of 6.02) that the most recent bit is on the left and the least recent on the right. We tried to be careful in giving full credit to students who had the reverse interpretation, *as long as it was consistent throughout the questions.*