CMIMC 2020
Power Round

INSTRUCTIONS

1. Do not look at the test before the proctor starts the round.
2. This test consists of several problems, some of which are short-answer and some of which require proofs, to be solved within a time frame of **60 minutes**. There are **70 points** total.
3. Answers should be written and clearly labeled on sheets of blank paper. Each numbered problem should be *on its own sheet*. If you have multiple pages, number them as well (e.g. 1/3, 2/3).
4. Write your team ID on the upper-right corner and the problem and page number of the problem whose solution you are writing on the upper-left corner on each page you submit. Papers missing these will not be graded. Problems with more than one submission will not be graded.
5. Write legibly. Illegible handwriting will not be graded.
6. In your solution for any given problem, you may assume the results of previous problems, even if you have not solved them. You may not do the same for later problems.
7. Problems are not ordered by difficulty. They are ordered by progression of content.
8. No computational aids other than pencil/pen are permitted.
9. If you believe that the test contains an error, submit your protest in writing to Doherty 2302 by the end of lunch.

Problems

In this power round, we will explore the mathematics of Coding Theory.

Contents

1 Introduction

Alice has moved to a new rent-controlled condo two thousand feet under the sea. Unfortunately, whenever she sends letters to her pen-pal Bob, the water dampens the letters and makes them hard to read. Specifically, their letters can be corrupted in two different ways:

- **Erasures:** The water obscures symbols in the transmitted messages. This can be understood as replacing them with '?'s.
- **Errors:** The water turns some of the symbols into other symbols.

Is it possible for Bob to error-proof his messages so that they can be decoded even if the water corrupts them? This simple question is the basis of "coding theory", which is an important and active area of computer science research today. In this power round, we will explore the mathematics of sending messages and apply them to some novel situations.

Formally, we define a message as follows:

Definition 1 (Alphabet)**.** An *alphabet* is a finite set of symbols Σ, such as $\{0, 1\}$ or $\{a, b, c\}$.

Definition 2 (Message)**.** A *message* is a sequence of symbols, selected from some alphabet Σ. The set of messages of length k over Σ is indicated as Σ^k. For example, $\{a, b\}^2 = \{aa, ab, ba, bb\}$.

For example, if Alice wants to send Bob the message "GOOD LUCK," the message with an erasure would be "GOO? LUCK," and the message with an error might be "GOOF LUCK."

2 Some Elementary Codes [18 points]

In this section, we shall devise some simple strategies to help Alice send messages to Bob in a way that allows him to detect and correct these two types of errors.

Definition 3 (Code)**.** A *(block) code* $\mathcal{C}$ is a subset of Σ^n, where n is called the *block length*. Elements of a code are called *codewords*.

Definition 4 (Encoding Map)**.** An *encoding map* $E : \Sigma^k \to \Sigma^n$ is an injective function from message set $\mathcal{M} = \Sigma^k$ to Σ^n (i.e. no two words in Σ^k map to the same word under E). The resultant code is the image of the map, i.e. $\mathcal{C} = E(\mathcal{M})$. The integer k is called the *message length*.

In order to send a message $m \in \mathcal{M}$ using a code, we instead transmit $E(m)$ over the channel. Thus, the goal is to devise codes and encoding maps so that we are able to decode m from a "corrupted" version of $E(m)$.

Problem 2.1 (Repetition Code, 2 points)

A simple way of handling corruption is to encode m by repeating each symbol of message m a certain number of times.

(i) (1 point) Suppose the channel can introduce up to t erasures. How many times should we repeat each symbol in the message to ensure that we can recover the original message?

(ii) (1 point) Repeat the previous problem, now assuming the channel can introduce up to t errors.

(i) Each symbol should be repeated $t + 1$ times.

(ii) Each symbol should be repeated $2t + 1$ times.

Definition 5 (q-ary Codes)**.** A code over the alphabet $\Sigma_q = \{0, 1, \ldots, q - 1\}$ is called a *q-ary* code. A 2-ary code is called a *binary* code.

Problem 2.2 (2 points)

Devise a code/encoding map with the smallest possible block length for sending the message set Σ_2^k across a channel which can introduce at most one erasure and no errors.

Append a parity bit: that is, let $p = \bigoplus_{i=1}^{k} s_i$ (the XOR of all the bits). Then the code size is $n = k+1$.

Problem 2.3 (6 points)
Devise a code/encoding map for sending the message set Σ_2^k across a channel which can introduce at most one error and no erasures. Your code should have block length $n \le 2k+1$. You may assume that k is sufficiently large. Points will be awarded based on how small the block length is.

Note that repetition codes would use $3n$ bits, which is insufficient for our purposes.
We give solutions achieving progressively better bounds.

- This solution achieves $\boxed{2k+1}$. Suppose the message is $s = s_1 s_2 \dots s_k$. Let $p = \bigoplus_{i=1}^{k} s_i$ (the XOR of all the bits) and send over ssp.

  This gets 3 points.

- This solution achieves $\boxed{k + \mathcal{O}(\sqrt{k})}$. Round k up to the nearest perfect square n^2. Now create an $n \times n$ matrix A with the entries being the bits of k, padded with extra 0s. For a row $\vec{A_i}$, let $p_i = \bigoplus_{j=1}^{n} A_{i,j}$ and for a column $\underline{A_i}$ let $q_i = \bigoplus_{j=1}^{n} A_{j,i}$.
  From this information, we create the new matrix

  $$A' = \left[\begin{array}{ccc|c} A_{1,1} & \cdots & A_{1,n} & p_1 \\ \vdots & \ddots & \vdots & \vdots \\ A_{n,1} & \cdots & A_{n,n} & p_n \\ \hline q_1 & \cdots & q_n & 0 \end{array}\right]$$

  Send the $(n+1)^2 - 1$ entries that are not the bottom right entry. The rounding incurs at most a $\mathcal{O}(\sqrt{k})$ cost and so does the overhead of the sent entries.

  This gets the full 6 points.

- This solution achieves $\boxed{k + \mathcal{O}(\log k)}$. Use a hamming code. Round k up to the nearest number of the form $2^r - r - 1$. We can do a parity check of size r to get a final code of size $2^r - 1$.

  We note, however, that the rounding may double the code size. To alleviate this, split the message into at most $\sqrt{k}(?)$ chunks which are close in size to $2^r - r - 1$. (Note: check later)

  This gets 8 points (two bonus).

While the codes above are easy solutions, they are not very efficient and thus not good solutions to Alice's problem. We now move toward a family of codes that are much more efficient (and actually implemented in practice!). To this end, we need a few definitions.

Definition 6 (Hamming Distance)**.** For two messages of equal length x and y, the *Hamming distance* $d(x, y)$ is the number of indices in the string where the messages differ. For example, $d(00221, 01011) = 3$.

Definition 7 (Hamming Weight)**.** For a string m, the *Hamming Weight* $\mathrm{wt}(m)$ is the number of symbols which are nonzero. For example, $\mathrm{wt}(012001) = 3$.

Definition 8 (Minimum Distance)**.** The *minimum distance* of a code $\mathcal{C}$ is defined as the smallest hamming distance between any two distinct codewords in $\mathcal{C}$. Formally,

$$d(\mathcal{C}) = \min_{x \neq y \in \mathcal{C}} d(x, y)$$

We also need the notion of a *linear code*.

Definition 9 (Linear Code)**.** A q-ary encoding map $E : \Sigma_q^k \to \Sigma_q^n$ is said to be linear if

- $E(0) = 0$

- For all $x, y \in \Sigma_q^k$, $E(x+y) = E(x) + E(y)$

where addition is element-wise modulo q. The resultant code is called a linear code.

Problem 2.4 (2 points)
Prove that for any q-ary linear code $\mathcal{C}$, the following relation holds.

$$d(\mathcal{C}) = \min_{c \in \mathcal{C}, c \neq 0} \operatorname{wt}(c)$$

Let $\Sigma = \{0, 1, \ldots, q-1\}$ and $E : \Sigma^k \to \Sigma^n$ be the encoding map corresponding to $\mathcal{C}$ We write $d(\mathcal{C}) = \min_{x,y \in \Sigma^k} d(E(x), E(y))$. However, note that

$$d(E(x), E(y)) = \operatorname{wt}(E(x-y)) = d(E(x-y), 0) \qquad \text{(linearity)}$$
$$\therefore \min_{x \neq y \in \Sigma^k} d(E(x), E(y)) = \min_{x \neq 0 \in \Sigma^k} d(E(x), 0) = \min_{x \neq 0} \operatorname{wt}(c)$$

Problem 2.5 (4 points)
Prove that the following statements for a code $\mathcal{C}$ are equivalent.

1. $\mathcal{C}$ has minimum distance at least $2t+1$.
2. $\mathcal{C}$ can be used to correct t symbol errors.
3. $\mathcal{C}$ can be used to correct $2t$ symbol erasures.

We prove $1 \implies 2, 3$, and that each of $2, 3$ imply 1.
Suppose 1 is true, and let m be a received message.

2. If m has at most t symbol errors, then there must exist a unique codeword c where $d(c, m) \leq t$. Hence m is uniquely decodable.
3. If m has at most $2t$ erasures, then suppose there are two codewords b, c which match up with m outside the erasures. Then their distance is at most $2t$, a contradiction.

Now we prove the backward implications.

2. Suppose $\mathcal{C}$ can correct t symbol errors and $d(\mathcal{C}) < 2t+1$. In particular, suppose $d(x, y) < 2t+1$ with $x, y \in \operatorname{im}(\mathcal{C})$. Then let m be such that it agrees with x, y everywhere they agree, and otherwise $d(x, m) = t$. Then this gives a contradiction.
3. We use the same setup as before. Let m agree with x and y everywhere the two agree, and put in ? everywhere they don't. Then this is uncorrectable.

Problem 2.6 (2 points)
Let the (7, 4) **Hamming code** be the 2-ary code obtained by encoding map

$$(x_1, x_2, x_3, x_4) \mapsto (x_1, x_2, x_3, x_4, x_1 \oplus x_2 \oplus x_4, x_1 \oplus x_3 \oplus x_4, x_2 \oplus x_3 \oplus x_4)$$

where $\oplus$ denotes addition modulo 2. Find, with proof, the minimal distance of this code.

We prove the minimal distance is 3. To do so, we exploit Problem 1.4 and case on the Hamming weight of the message x.

- If $\mathrm{wt}(x) = 0$, then x is 0 so it is not considered in the minimum.
- If $\mathrm{wt}(x) = 1$, then note that at least two parity bits are set. So, the weight of the codeword is at least 3.
- If $\mathrm{wt}(x) = 2$, then note that at least one parity bit is set so addition yields 3.
- If $\mathrm{wt}(x) \geq 3$, then we are already done.

Hence, the minimal distance is at least 3. It is easy to see that

$$(1,0,0,0) \mapsto (1,0,0,0,1,1,0)$$

so the distance is at most 3 as well.

Remark. The above construction can be generalized for larger values of 7 and 4.

3 Bounds on Code Size [12 points]

In devising codes, there is always a trade-off between message length (how much we can transmit) and code distance (how much we can correct). In this section, we will derive a few bounds quantifying this trade-off.

Problem 3.1 (Singleton Bound, 3 points)
Let $\mathcal{C}$ be a q-ary code with block length n and minimum distance d. Prove that

$$|\mathcal{C}| \leq q^{n-d+1}.$$

Assume for the sake of contradiction, $|\mathcal{C}| > q^{n-d+1}$. By pigeonhole principle, there must be two distinct codewords $c_1, c_2 \in \mathcal{C}$ that agree on the first $n-d+1$ locations. But then $d(c_1, c_2) \leq d-1$ contradicting the hypothesis that C has minimum distance d.

Problem 3.2 (4 points)
Let $\mathcal{C}$ be a binary code with block length n and minimum distance 3. Prove that

$$|\mathcal{C}| \leq \frac{2^n}{n+1}.$$

For each codeword $c \in \mathcal{C}$, consider the Hamming ball of radius 1 centered at c, $B(c) = \{y \in \{0,1\}^n : d(y,c) \leq 1\}$. Clearly $|B(c)| = n+1$. If for some two codewords $c, c' \in \mathcal{C}$, we have $B(c) \cap B(c') \neq \emptyset$ then $d(c,c') < 3$ and therefore $c = c'$. So no two such balls intersect. So

$$2^n \geq \left| \bigcup_{c \in \mathcal{C}} B(c) \right| = \sum_{c \in \mathcal{C}} |B(c)| = 2^k(n+1)$$

$$\therefore 2^k \leq \frac{2^n}{n+1}$$

Problem 3.3 (Hamming Bound, 3 points)

Let $\mathcal{C}$ be a q-ary code with block length n and minimum distance d. Generalize the previous problem to prove that, for $t = \lfloor \frac{d-1}{2} \rfloor$,

$$|\mathcal{C}| \leq \frac{q^n}{\sum_{i=0}^{t} \binom{n}{i} (q-1)^i}.$$

For every codeword $c \in \mathcal{C}$, consider the Hamming ball of radius t centered at c. Notice that $|B(c)| = \sum_{i=0}^{t} \binom{n}{i} (q-1)^i$. This is because every $x \in B(c)$ differs from c in at most t coordinates, so we sum over these and choose the symbol which x takes on at that position. Now we finish similarly to the previous part.

Problem 3.4 (2 points)

Using the previous problem, prove that, for any $1 \leq d \leq k$, any binary encoding map with message length k and minimum distance d has block length $n \geq k + \frac{d-2}{2} \log_2(\frac{k}{d})$. You may use the fact that $\binom{n}{t} \geq \left(\frac{n}{t}\right)^t$ for any positive integers n and t with $n \geq t$.

By the Hamming Bound, any code $\mathcal{C}$ has $|\mathcal{C}| \leq \frac{2^n}{\sum_{i=0}^{t} \binom{n}{i}} \leq \frac{2^n}{\binom{n}{t}} \leq 2^n \left(\frac{t}{n}\right)^t$. So, $\mathcal{C}$ has message length $k = \log_2 C = n + t \log_2 \left(\frac{t}{n}\right)$. So $n \geq k + t \log_2 \frac{n}{t} \geq k + \frac{d-2}{2} \log_2 \left(\frac{k}{d}\right)$.

4 Reed-Solomon Codes [9 points]

In this section, we shall now develop a family of codes for Alice and Bob to use, which uses small block length relative to message length.

It turns out that a $(k-1)$ degree polynomial can be uniquely determined from its evaluation at any k points. So, to allow for t erasures, we will transform a message m into a $(k-1)$ degree polynomial and encode it as the evaluation of that polynomial at $k+t$ different points. Then, if any t values in the codeword are lost, we can still recover m from the remaining k evaluation points.

We shall now formalize the above argument. Let q be prime and $q \geq n = k + t$

Now consider the following algorithm, given an input m

1. Let $m = m_1 \ldots m_k$, and find $p(x) = m_1 + m_2 x + \cdots + m_k x^{k-1}$.
2. Compute $b_i = p(i) \mod q$ for $i = 1, \ldots, n$.
3. Return $b_1 \ldots b_n$.

The above is an encoding map $E_{RS} : \Sigma^k \to \Sigma^n$; let $\mathcal{C}_{RS}$ be the associated code. First, we shall find the minimum weight of $\mathcal{C}_{RS}$.

Problem 4.1 (Lagrange's Theorem, 2 points)

Let q be prime. Show that for any polynomial $p(x)$ with integer coefficients not all divisible by q, $p(x) \equiv 0 \mod q$ for at most $\deg(p)$ distinct values of x.

We proceed by induction; it is easy to verify for $\deg(p) = 0$. Otherwise, suppose $p(a) \equiv 0 \mod q$; then we can write $p(x) = (x-a)r(x) + s$. Then $s \equiv p(a) \equiv 0 \mod q$. Not all coefficients of r can be multiples of q, so by induction, $r(x) \equiv 0$ for at most $\deg(p) - 1$ values of x. Since $p(x) \equiv (x-a)r(x) \mod q$, we see that $p(x) \equiv 0 \mod q$ for at most $\deg(p)$ values of x. completing the induction.

Problem 4.2 (2 points)

Show that $d(\mathcal{C}_{RS}) = n - k + 1$.

Let p_1 be the polynomial created from m_1 and p_2 the polynomial created from m_2. Then $p_1 - p_2$ is a nonzero polynomial, so $(p_1 - p_2)(x) \equiv 0 \mod q$ for at most $k-1$ values of x by the previous problem. So $p_1(x) \equiv p_2(x) \mod q$ for at most $k-1$ values of x. So $C(m_1)$ and $C(m_2)$ must differ for at least $n-k+1$ values, so $d(\mathcal{C}_{RS}) = n-k+1$. By the Singleton bound, we must have equality.

Using problem 1.5, we know that we can use a Reed Solomon Code to detect erasures and errors efficiently. But we still need to actually figure out an algorithm to do this.

Problem 4.3 (2 points)

Show that, for any $x_1, x_2, x_3, \ldots, x_k$ and any y, we can find a polynomial f of degree $k-1$ with integer coefficients so that $f(x_1) \equiv y \mod q$ and $f(x_i) \equiv 0 \mod q$ for $2 \leq i \leq k$.

First consider the polynomial $g(x) = (x - x_2) \cdots (x - x_k)$; this polynomial has degree $k-1$ and goes through $(x_i, 0)$ for all $2 \leq i \leq s$. Now let $g(x_1) = a$; then there exists some element b so that $ab = 1 \mod q$. Then $ybg(x)$ goes through all the desired points.

Problem 4.4 (3 points)

Devise an algorithm that allows you to, given the code $E_{RS}(m)$ of length $k+t$ with at most t erasures, decipher the original length k message m.

Let $I \subseteq [k+t]$ represent the indices of the first k unerased symbols.Then we want to find the polynomial p so that $p(i) \equiv b_i \mod q$ for all $i \in I$. By problem 3, we can get p_i so that $p_i(i) \equiv b_i \mod q$, and $p_i(j) \equiv 0$ for $j \in I \backslash \{i\}$. Then, adding up these polynomials $\mod q$, we get a polynomial p so that $p(i) \equiv b_i \mod q$ for all i.

5 LT Codes [19 points]

Using the codes devised in sections 1 and 3, Alice and Bob have now devised an effective method of correcting for erasures and errors. Unfortunately, due to pollution, the water will now simply destroy $\frac{1}{2}$ of the messages Alice sends.

Alice wants to send Bob n messages, all of the same length. When Bob's received the n messages, he will send her a special passage in response. Unfortunately, Alice has no way to know whether her messages reach Bob, so they have to adopt a clever strategy to ensure the expected number of messages Alice must send is small.

Problem 5.1 (3 point)

One strategy would be for Alice to send all n messages one after another, then wait to see whether Bob has received them all, and send them all again if not. Bob will keep every message he receives until he has eventually received all of them. Show that, for sufficiently large n, the expected number of messages Alice sends is more than $10n$.

The probability that a message has made it through after r cycles is $1-\frac{1}{2^r}$. So the probability that all messages have made it through is $(1-\frac{1}{2^r})^n$. So, if X is the number of times Alice must cycle through, then $P[X>r] = 1-(1-\frac{1}{2^r})^n$, so $E[X] = \sum_{r=0}^{\infty}(1-(1-\frac{1}{2^r})^n)$. So $E[X] \geq \sum_{r=0}^{11}(1-\left(\frac{2047}{2048}\right)^n) = 12(1-\left(\frac{2047}{2048}\right)^n)$. So for sufficiently large n, the expected number of messages Alice must sent is at least $10n$.

Suppose that Alice's messages are $m_1, \ldots, m_n$. Suppose that Alice picks a random binary string s of length n, and then constructs a packet m_s by $\oplus$ing every message for which s has a 1. For example, if $n=4$, $m_{1010} = m_1 \oplus m_3$. Each turn, Alice will generate a random s and send Bob the packet m_s.

Problem 5.2 (1 point)

Let e_i be the binary string of length n with a 1 at the $i$the entry and 0s elsewhere. Show that, if she can write $e_i = s_1 \oplus \cdots \oplus s_k$, then $m_i = m_{s_1} \oplus \cdots \oplus m_{s_k}$.

Note it is sufficient to show that $m_{s_1 \oplus s_2} = m_{s_1} \oplus m_{s_2}$. Clearly the statement is true if s_1, s_2 share no 1s at common indices. Let z contain the bits which are in both s_1 and s_2, and let $s_1 = t_1 \oplus z$ and $s_2 = t_2 \oplus z$. Then $m_{s_1} = m_z \oplus m_{t_1}$ and $m_{s_2} = m_z \oplus m_{t_2}$. So $m_{s_1} \oplus m_{s_2} = m_z \oplus m_{t_1} \oplus m_z \oplus m_{t_2} = m_{t_1} \oplus m_{t_2} = m_{t_1 \oplus t_2} = m_{s_1 \oplus s_2}$.

The span of a set of binary strings S is the set of strings which can be formed by $\oplus$ing some of them together. In other words, $\text{span}(S) = \{\bigoplus_{s \in T} s : T \subseteq S\}$. Let $\dim(S) = \log_2(|\text{span}(S)|)$

Problem 5.3 (3 points)

Show that, if $t \in \text{span}(S)$, then $\dim(S \cup \{t\}) = \dim(S)$; otherwise, if $t \notin \text{span}(S)$, then $\dim(S \cup \{t\}) = \dim(S)+1$. Also show that, if S contains enough packets for Bob to decode all her original messages, then $\dim(S) = n$.

If $t \in \operatorname{span}(S)$, then you can write $t = s_{i_1} \oplus \cdots \oplus s_{i_d}$. Then any element of $s_1 \oplus \cdots \oplus s_d \oplus t = s_1 \oplus \cdots \oplus s_d \oplus s_{i_1} \oplus \cdots \oplus s_{i_d}$, which will reduce to a $\oplus$ of various elements of S, so $\operatorname{span}(S) = \operatorname{span}(S \cup \{t\})$. Otherwise, every element of $s_{i_1} \oplus \cdots \oplus s_{i_d} \oplus t$ is unique, since if $s_{i_1} \oplus \cdots \oplus s_{i_a} \oplus t = s_{j_1} \oplus \cdots \oplus s_{j_b} \oplus t$, then $s_{i_1} \oplus \cdots \oplus s_{i_a} \oplus s_{j_1} \oplus \cdots \oplus s_{j_b} = 0$. So $\operatorname{span}(S \cup \{t\})$ has twice has many elements as $\operatorname{span}(S)$, so $\dim(S \cup \{t\}) = \dim(S) + 1$.

Suppose that Alice sends Bob the packets $m_{s_1}, m_{s_2}, \ldots$. Let $d_i = \dim(\{s_1, \ldots, s_i\})$, with $d_0 = 0$.

Problem 5.4 (10 points)

Alice begins sending packets to Bob.

(i) (5 points) Find the exact probability that Bob can recover all n original messages after n packets, and show that this probability is at least $\frac{1}{4}$ for all n.

(ii) (5 points) Show the probability that Bob can recover the n message after $n+2$ packets is at least $\frac{3}{5}$ for all $n \geq 5$.

- Note that d_{i+1} will be $d_i + 1$ with probability $1 - \frac{2^{d_i}}{2^n}$, and stay the same otherwise. For Bob to able to recover the message after he receives n packets, the d_i must increase every time. This has probability $\prod_{i=0}^{n-1} \left(1 - \frac{2^i}{2^n}\right)$. Then note that

$$\prod_{i=0}^{n-1} \left(1 - \frac{2^i}{2^n}\right) = \prod_{k=1}^{n} \left(1 - \frac{1}{2^k}\right) =: S_n$$

We shall show via induction that $S_n \geq \frac{1}{4} + \frac{1}{2^{n+1}}$. Clearly this is true for $n = 1$; moreover,

$$\begin{aligned} S_{n+1} = S_n \left(1 - \frac{1}{2^{n+1}}\right) &\geq \left(\frac{1}{4} + \frac{1}{2^{n+1}}\right)\left(1 - \frac{1}{2^{n+1}}\right) \\ &= \frac{1}{4} + \frac{1}{2^{n+1}} - \frac{1}{2^{n+3}} - \frac{1}{2^{2n+2}} \geq \frac{1}{4} + \frac{1}{2^{n+2}}. \end{aligned}$$

- The d_i will have to increase every time but one. The probability that she increases every time except the jth and kth is $\frac{2^{j-1}}{2^n}\frac{2^{k-2}}{2^n}\prod_{i=0}^{n-1}\left(1 - \frac{2^i}{2^n}\right) \geq \frac{2^{j-3}}{2^n}$. So the probability that she can recover the message after $n+2$ packets is at least

$$\frac{1}{4}\sum_{j=1}^{n+1}\sum_{k=j+1}^{n+2} \frac{2^{j-1}}{2^n}\frac{2^{k-2}}{2^n} = \frac{1}{4}\sum_{j=0}^{n}\sum_{k=j}^{n} \frac{2^j}{2^n}\frac{2^k}{2^n} = \frac{1}{4}\left(\frac{8}{3} - \frac{1}{2^{(n-1)}} + \frac{1}{3 \cdot 2^{2n}}\right) \geq \frac{3}{5}$$

for $n \geq 5$.

Problem 5.5 (2 points)

Show that the expected number of packets Alice must send until Bob has enough to decode all n messages is at most $8n$ for sufficiently large n. Remember that $\frac{1}{2}$ of the messages are lost.

If Alice sends $2(n+1)$ messages, the probability that at least $n+1$ make it through is more than $\frac{1}{2}$, so the probability that that $2(n+1)$ messages is sufficient for Bob is at least $\frac{3}{10}$. So we have a geometric series that will take in expectation at most $\frac{10}{3}$, so the expected number of total messages to be sent will be at most $\frac{20}{3}(n+1) \leq 8n$ for sufficiently large n.

6 Miscellaneous [12 points]

Now that Alice and Bob have figured out how to communicate, they are free to have fun working on random coding theory problems.

Problem 6.1 (4 points)

Consider a code with variable length $\mathcal{C} = \{1, 01, 001, 0001\}$ in which each of the codewords are transmitted with probabilities $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}$ respectively. What is the probability that a randomly chosen bit in a long stream of transmission is a 1?

We compute the expected number of ones per bit transmitted.
Codeword $c \in C$ is transmitted with probability $p(c)$. It adds $\text{len}(c)$ bits to the stream, of which $\text{wt}(c)$ bits are ones. So every time a new codeword is transmitted, the expected number of bits added to the stream is $\sum p(c)\text{len}(c)$, while the expected number of ones added is $\sum p(c)\,\text{wt}(c)$. Therefore, the fraction of ones in the transmitted string is

$$\frac{\sum p(c)\,\text{wt}(c)}{\sum p(c)\text{len}(c)} = \frac{\frac{1}{2}\cdot 1 + \frac{1}{4}\cdot 1 + \frac{1}{8}\cdot 1 + \frac{1}{8}\cdot 1}{\frac{1}{2}\cdot 1 + \frac{1}{4}\cdot 2 + \frac{1}{8}\cdot 3 + \frac{1}{8}\cdot 4} = \frac{8}{15}$$

(Note, the different codewords contribute different numbers of bits to the stream, so their ones per bit contribution must be weighted by their string length.)

Problem 6.2 (8 points)

Suppose that n people are dealt a red or black card, each independently with probability $\frac{1}{2}$. Each person holds up their card so that everyone can see it but themselves. Each person is given a chance to guess their own card, or to pass. The group wins as long as someone can guess which color card they have, and no one guesses wrong. Otherwise, they lose.

(i) (1 point) Give a strategy where the group wins with probability $\frac{1}{2}$.

(ii) (7 points) Suppose $n = 7$. Give a strategy where the group wins with probability $\frac{7}{8}$.

Consider the 7 symbol message formed from the red and black cards; each person can see 6 of them, so each person can see a message with one error. Consider the (7,4) Hamming Code, and suppose each person guesses the hat color that would make the sequence not a codeword, and nothing otherwise. Note that no binary string of length 7 is within distance 1 of two codewords, and since each codeword as 7 other codewords within distance 1, every 7 symbol string is either a codeword or distance 1 to a unique codeword, with probability $\frac{1}{8}$ and $\frac{7}{8}$ respectively. If the string is in fact a codeword, then everyone will guess wrong and the group loses. Otherwise, only one person guesses, and they will be right so the group will win.