Ilana Sivan
205634272
Honors Course
Homework 8

Question 1

Prove that the Hamming distance is a metric. That is to say, for all $x, y, z \in \{0,1\}^n$

1. $\Delta(x,y) \geq 0$

   Since the Hamming distance is a measure of the number of positions in which two vectors differ, this number is clearly bounded from below by zero (cannot differ in -1 positions). So $\Delta(x,y) = |\{i \in [n] : x_i \neq y_i\}| \geq 0$

2. $x = y \Leftrightarrow \Delta(x,y) = 0$

   If $\Delta(x,y) = 0$, then x and y do not differ in any positions, that is to say that $x = y$. Similarly, if $x = y$, then x and y do not differ in any positions, that is $\Delta(x,y) = 0$. So, $\Delta(x,y) = |\{i \in [n] : x_i \neq y_i\}| = |\emptyset| = 0 \Leftrightarrow x = y$.

3. $\Delta(x,y) = \Delta(y,x)$

   Since the number of positions in which two vectors differ is not dependent on the order in which the vectors are considered, this is trivially true. So, $\Delta(x,y) = |\{i \in [n] : x_i \neq y_i\}| = |\{i \in [n] : y_i \neq x_i\}|\Delta(y,x)$

4. $\Delta(x,y) + \Delta(y,z) \geq \Delta(x,z)$

   We can show this by constructing sets, and comparing their cardinalities.

   Let $S_1 = \{i | x_i \neq z_i\}$

   Let $S_2 = \{i | x_i \neq y_i\}$

   Let $S_3 = \{i | y_i \neq z_i\}$

   Clearly $|S_1| = \Delta(x,z), S_2 = \Delta(x,y)$. If we show that $|S_1| \leq |S_2| + |S_3|$, we will succeed in showing that $\Delta(x,y) + \Delta(y,z) \geq \Delta(x,z)$. For arbitrary i, we can check which of the above sets contain i.

   - Suppose that $x_i \neq z_i$.
     Then $i \in S_1$ and i also belongs to one or both of S_2, S_3. Then, i appears at least once on the right hand side if $x_i = y_i$ or $z_i = y_i$ if not twice.
   - Suppose that $x_i = z_i$.
     Then $i \notin S_1$.However, if $y_i \neq x_i = z_i$ then $i \in S_2, i \in S_3$. In either case, we do not include i on the left hand side, but may include it on the right hand side up to two times.

   We have seen that all elements from the left hand side also exist in the right hand side, that is to say that $|S_1| \leq |S_2| + |S_3|$, so $\Delta(x,y) + \Delta(y,z) \geq \Delta(x,z)$.

Question 2

Prove that all of the following statements are valid for the code $C \subseteq \Sigma^n$

1. The minimal distance of C is d

   $d_H = (u,v)$, the Hamming distance between two vectors u, v is the number of coordinates by which they differ, that is the Hamming distance between two vectors $u, v \in \{0,1\}^n$ is defined to be the number of coordinates that do not match. In other words, we have $d_H(u,v) = w_H(u+v \mod 2)$, and $u+v \bmod 2$ is a boolean XOR or coordinate wise sum modulo 2. We can also observe this as a minimum pairwise Hamming distance between any two code words: $d_H(C) = \min_{u,v \in C : u \neq C} d_H(u,v)$. The weight $wt_H(u)$ is the number of coordinates which are non-zero. So we get $d_H(u,0) = wt_H(u)$ where 0 is the all-zero vector of length n. The distance between u, v is the same as the weight of their difference then: $d_H(u,v) = wt_H(u-v)$. For a code C that contains at least two code words the minimum weight of C then is the smallest weight of all the nonzero codewords in C, and the smallest distance between any two distinct codewords in C is the minimum Hamming distance of C. That is that as $d_H(u,v) = wt_H(u-v)$ that we have a minimum distance of a code is $wt_H(u,v)$ and $u \neq v$. So for linear codes the minimum weight is the same as the minimum distance.

2. If d is odd then C can correct $\frac{(d-1)}{2}$ errors.

   C code of length n is s-error detecting if changing up to s digits to a codeword does not produce a codeword and t-error correcting if from a string of length n that differs on up to t places from some codeword we can find the codeword. We can detect up to s errors in any codeword if $d(C) \geq s+1$ and we can correct up to t errors in any codeword if $d(C) \geq 2t+1$. We can prove this as follows: Let $d(C) \geq s+1$. A codeword u is transmitted and we have up to $s$errors, in this case the received vector cannot be a different codeword and the errors can be detected. Let $d(C) \geq 2t+1$. A codeword u is sent and the vector v is received in which we have t or fewer errors, so we get that $d(u,v) \leq t$. Let $z$be a codeword with $d(z,v) \leq t$. Then $d(u,z) \leq d(u,v) + d(v,z) \leq 2t$, hence $u = z$. So u is the nearest codeword to v. Now we can show that C can correct $\frac{(d-1)}{2}$ errors: Let C be a code with minimum distance d. Then up to $d-1$ errors can be detected and up to $\frac{(d-1)}{2}$ errors can be corrected because $d \geq s+1$ if and only if $s \geq d-1$, and $d \geq 2t+1$ if and only if $t \leq \frac{(d-1)}{2}$.

3. C can identify $d-1$ errors

   Let d be the smallest Hamming distance between two codewords in a code C, then $C = d_H(C) = \min_{u,v \in C: u \neq C} d_H(u,v)$. If we want to change one codeword to another we need at least d bit changes. So C can detect up to $d-1$ errors, because $d-1$ transmission errors cannot change one codeword to another.

   If we have some codeword $u \in C$ that is transmitted and we have some r errors, and we receive v, we can use the following to identify an error:

   $$Identify(v) = \begin{cases} u' & \exists u' \in C st. u' = v \\ error & else \end{cases}$$

   If there is an undetectable error, then we have $u' = v$ and $u' \neq u$. Which is to say that $r = d_H(u,v) = d_H(u,u') \geq d$.

   So the code can detect $d-1$ errors.

4. C can recover $d-1$ erasures

   Trivially, since any two codewords are a distance of d apart, you can differentiate them after d−1 erasures, as there will still be at least a coordinate where they differ.

We show that if 1 is not satisfied none of the others are.

$\neg 1 \rightarrow \neg 2$: Let $c_1 \neq c_2 \in C$ be codeworsd such that $\Delta(c_2, c_1) = d-1$, now for a vector a such that $\Delta(a, c_1) = \Delta(a, c_2) = \frac{d-1}{2}$. This a exists because d is odd by choosing c_1, c_2. We could get a if either c_1, c_2 were transmitted so you cannot decode.

$\neg 1 \rightarrow \neg 3$: Let c_1, c_2 be codewords such that $\Delta(c_2, c_1) = d-1$. Let $a = c_2$, then either we detect no algorithm or we detect an error when c_2 is the codeword that is transmitted and no error takes place during its transmission.

$\neg 1 \rightarrow \neg 4$: For a word a in which the positions erased are where c_1, c_2 differ, we get that $c_1$and c_2 could hav ebeen transmitted and we cannot correct $d-1$ erasures.

Question 3

Prove that for every binary linear code $C \subseteq \{0,1\}^n$ that the minimum distance is equal to the weight of the Hamming distance of a code word with minimal hamming weight.

0 is a code word for any linear code C. First we show that every weight is a distance, as $wt(c) = d(c,0)$, every weight is a distance. Conversely, $d(c1, c2) = wt(c1 - c2)$, and, since C is linear, $c1 - c2 \in C$, so every distance is a weight. Let c be be the codeword with minimum $wt(c)$. Notice, $wt(c) = \Delta(c,0)$. Then, by the denition of minimum distance, $d \leq \Delta(c,0)$. Alternatively, $d \leq wt(c)$. As linear codes have an additive inverse, we let $c_1 \neq c_2 \in C$ such that $\Delta(c_1, c_2) = d$ (must exist by denition of d). Since C is linear, $-c_2 = -1 * c_2 \in C$ and $c_1 + (-c_2) = c_3 \in C$, since $c_1 \neq c_2, c_3 \neq 0$. $wt(c_3) = \Delta(c_1, c_2)$ as only positions in which c_1 and c_2 match will be equal to 0 (positions in which they differ will be different from 0 in c_3). We now have that $wt(c_3) = d$. Therefore, $wt(c) \leq d$ where c has the minimum Hamming weight. Combining these two facts, the distance of C is equal to the smallest Hamming weight of any nonzero codeword.

Question 4

Prove that the minimal distance of the Hamming code $E_H : \{0,1\}^4 \rightarrow \{0,1\}^7$ defined as:

$E_H(x_1, x_2, x_3, x_4) = (x_1, x_2, x_3, x_4, x_1 \bigoplus x_2 \bigoplus x_4, x_1 \bigoplus x_3 \bigoplus x_4, x_2 \bigoplus x_3 \bigoplus x_4)$

is 3.

We assume that E_H is a linear code, which means that $E_H(x+y) = E_H(x)+E_H(y)$ for any inputs $x, y \in \{0,1\}^4$, and that the all-zero vector 0 is a codeword.

Let's enumerate the possible weights of x where x is a vector we input.

- If $wt(x) = 0$:

  In this case, x must be equal to 0, and so the proposition holds and we do not need to check it

- If $wt(x) = 1$:

  $|\{i \in [y] | x_i = 1\}| = 1$. In this case, a single data bit and two parity bits will be asserted in the codeword (each x_i affects two parity bits). If three bits are asserted, then $wt(C(x)) = 3$. As $c_i = x_i$ for $i \in [y]$ the single weight will contribute 1 towards $wt(c)$. As we have $x_i \oplus x_j = 1, \forall i \neq j$ and as each weight from $x$is included in at least two of the last three components of c as part of a XOR operation we have $wt(c) \geq 2 + 1 = 3$

- If $wt(x) = 2$:

  At least one parity bit is 1 will be asserted in the codeword and we have $wt(c) \geq 2 + 1 = 3$

- If $wt(x) \geq 3$:

  At this point and beyond, $wt(x) \geq 3$, therefore $wt(C(x)) \geq 3$ as every data bit in the input vector also appears in the corresponding codeword, so we're done.