Exercises to Chapter 2 (answers at end)

Exercise 2.1. The Manchester code was first used in Manchester Mark 1 computer at the University of Manchester in 1949 and is still used in low-speed data transfer (e.g. TV remote sending signals via infrared). This binary code consists of two codewords: 10 and 01. The codeword 10 is interpreted by the recipient as the message 0, and 01 is understood to mean 1; whereas the received word 00 or 11 indicates a detected error.

The following error-free fragment of a bit stream encoded by Manchester code (that is: the stream is a sequence of codewords) had been intercepted: ...

...010101x01011010...

What was the bit x?

Exercise 2.2. Consider the alphabet $\mathbb{Z}_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. The Luhn checksum of a word $x_1 x_2 \ldots x_{16} \in (\mathbb{Z}_{10})^{16}$ is $\pi(x_1) + x_2 + \pi(x_3) + x_4 + \pi(x_5) + \cdots + x_{16} \mod 10$, viewed as an element of $\mathbb{Z}_{10}$. Here $\pi: \mathbb{Z}_{10} \to \mathbb{Z}_{10}$ is defined by the rule "$\pi(a)$ is the sum of digits of $2a$". The Luhn code consists of all words in $(\mathbb{Z}_{10})^{16}$ whose Luhn checksum is 0.

(i) Write down all values of $\pi$ and check that $\pi$ is a permutation of the alphabet $\mathbb{Z}_{10}$.

(ii) Find the total number of codewords of the Luhn code.

(iii) Prove that a single digit error is detected by the Luhn code.

(iv) Look at your 16-digit debit/credit card numbers. Are they codewords of the Luhn code? If you have a card with a number which is not a codeword of the Luhn code, can you bring it to the tutorial? Thanks!

Exercise 2.3. (loosely based on question A4 from the January 2013 exam). Alice transmitted the same binary word of length 6 to Bob three times, but Bob received three different words: 101010, 011100, 110001. Engineer Clara told Bob that at most two bit errors could have occurred in each word during transmission. Help Bob to recover the word transmitted by Alice.
Exercise 2.4. Recall that the binary alphabet is $\mathbb{F}_2 = \{0, 1\}$. Let the code $\Sigma \subseteq \mathbb{F}_2^7$ consist of the following eight words:

- $0 0 0 0 0 0 0$,
- $1 0 0 1 1 1 0$,
- $0 1 0 0 1 1 1$,
- $1 0 1 0 0 1 1$,
- $1 1 0 1 0 0 1$,
- $1 1 1 0 1 0 0$,
- $0 1 1 1 0 1 0$,
- $0 0 1 1 1 0 1$.

Find the Hamming distances between all pairs of codewords. Suppose that a codeword of $\Sigma$ is transmitted, and one, two or three bit errors occur; show that the received word is not a codeword (i.e., the error is detected).

What is $\text{DECODE}(0001110)$? Give an example of a word which has more than one nearest neighbour in $\Sigma$. Try to see if there are words with three, four etc. nearest neighbours. Try to write a word with a maximum possible number of nearest neighbours in $\Sigma$.

Exercise 2.5. Look again at the Manchester code, the Luhn code and the code $\Sigma$ introduced in the previous exercises. For each of these codes, determine the parameters $[n, k, d]_q$ of the code; state how many errors can the code detect and how many errors can the code correct.

Exercise 2.6. (a) Let $C$ be a $q$-ary code of length $n$. Denote $M = |C|$. Show: $n \geq \log_q M$.

(b) Assume that the cost of transmitting one symbol via a $q$-ary channel is $cq$. (Imagine a $q$-ary channel as a cable with $q$ wires; the costs of building and maintaining it would be roughly proportional to $q$.) Suppose that you need to design a code with $M \gg 0$ codewords (that is: $M$ is a very large number), and you have the control over the length $n$ and the size $q$ of the alphabet. Which $q$ will ensure the lowest transmission costs per codeword? In particular, are the binary channels (the type most widely used in today’s computer networks) the most economical?
Exercises to Chapter 2 — solutions

Exercise 2.1. The Manchester code was first used in Manchester Mark 1 computer at the University of Manchester in 1949 and is still used in low-speed data transfer (e.g. TV remote sending signals via infrared). This binary code consists of two codewords: 10 and 01. The codeword 10 is interpreted by the recipient as the message 0, and 01 is understood to mean 1; whereas the received word 00 or 11 indicates a detected error.

The following error-free fragment of a bit stream encoded by Manchester code (that is: the stream is a sequence of codewords) had been intercepted: ...010101x01011010...

What was the bit x?

Answer to E2.1. In ...010101x01011010..., notice that 11 cannot be a codeword. Therefore, the bit stream is split into codewords in the following way:

...0|10|10|x|01|01|10|10|...

The codeword 1x must be 10 so x = 0.

Exercise 2.2. Consider the alphabet $\mathbb{Z}_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. The Luhn checksum of a word $x_1 x_2 \ldots x_{16} \in (\mathbb{Z}_{10})^{16}$ is $\pi(x_1) + x_2 + \pi(x_3) + x_4 + \pi(x_5) + \cdots + x_{16} \mod 10$, viewed as an element of $\mathbb{Z}_{10}$. Here $\pi: \mathbb{Z}_{10} \to \mathbb{Z}_{10}$ is defined by the rule “$\pi(a)$ is the sum of digits of $2a$”. The Luhn code consists of all words in $(\mathbb{Z}_{10})^{16}$ whose Luhn checksum is 0.

(i) Write down all values of $\pi$ and check that $\pi$ is a permutation of the alphabet $\mathbb{Z}_{10}$.

(ii) Find the total number of codewords of the Luhn code.

(iii) Prove that a single digit error is detected by the Luhn code.

(iv) Look at your 16-digit debit/credit card numbers. Are they codewords of the Luhn code? If you have a card with a number which is not a codeword of the Luhn code, can you bring it to the tutorial? Thanks!
Answer to E2.2. (i) $\pi$ is the following permutation of $\mathbb{Z}_{10}$:

\[
\begin{pmatrix}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
0 & 2 & 4 & 6 & 8 & 1 & 3 & 5 & 7 & 9
\end{pmatrix}.
\]

(ii) Every sequence of 15 digits is the beginning of exactly one Luhn codeword. Indeed, let $x_1, \ldots, x_{15} \in \mathbb{Z}_{10}$ be arbitrary. Calculate $z = \pi(x_1) + x_2 + \pi(x_3) + x_4 + \pi(x_5) + \cdots + \pi(x_{15})$. Then the one and only Luhn codeword of the form $x_1x_2\ldots x_{15}x_{16}$ is determined by $z + x_{16} \equiv 0 \mod 10$. This is the same as $x_{16} \equiv (-z) \mod 10$.

Therefore, the number of Luhn codewords is equal to the number of sequences of 15 digits, that is, $10^{15}$.

(iii) If $x_i$ is replaced by $y_i$, then the Luhn checksum changes by $y_i - x_i \mod 10$ (if $i$ is odd) or by $\pi(y_i) - \pi(x_i) \mod 10$ (if $i$ is even). In any case, if $y_i \neq x_i$, then neither of these changes is zero mod 10, hence altering a single digit changes the Luhn checksum. A codeword has Luhn checksum 0, hence changing a single digit in a codeword gives a word with non-zero Luhn checksum, i.e., not a codeword.

Exercise 2.3. (loosely based on question A4 from the January 2013 exam). Alice transmitted the same binary word of length 6 to Bob three times, but Bob received three different words: 101010, 011100, 110001. Engineer Clara told Bob that at most two bit errors could have occurred in each word during transmission. Help Bob to recover the word transmitted by Alice.

Answer to E2.3. Let $\mathbf{z}$ be the word sent by Alice. We are given that each one of the words

$\mathbf{v}_1 = 101010, \quad \mathbf{v}_2 = 011100, \quad \mathbf{v}_3 = 110001$

received by Bob contains at most two errors. To work out $\mathbf{z}$, let’s try to see where the errors in $\mathbf{v}_1, \mathbf{v}_2$ and $\mathbf{v}_3$ could have occurred. Write $\mathbf{v}_1, \mathbf{v}_2$ and $\mathbf{v}_3$ as rows of a matrix:

\[
\begin{pmatrix}
1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1
\end{pmatrix}
\]

Note that each column of the matrix contains both 0s and 1s which cannot both be correct. Hence each column contains at least one error, and in total, there are at least 6 errors in the whole matrix. On the other hand, we are given that there were at most $6 = 2 + 2 + 2$ errors. Thus, there are exactly 6 errors in the matrix, and the only $\mathbf{z}$ which guarantees 6 errors is the word which ensures that there is exactly one error per
column — that is, the “majority” bits in each column must be correct:

\[
\begin{array}{cccccc}
1 & 0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 0 & 1
\end{array}
\]
\[
\hat{z} = 1 1 1 0 0
\]

**Exercise 2.4.** Recall that the binary alphabet is $\mathbb{F}_2 = \{0, 1\}$. Let the code $\Sigma \subseteq \mathbb{F}_2^7$ consist of the following eight words:

- 0 0 0 0 0 0 0
- 1 0 0 1 1 1 0
- 0 1 0 0 1 1 1
- 1 0 1 0 0 1 1
- 1 1 0 1 0 0 1
- 1 1 1 0 1 0 0
- 0 1 1 1 0 1 0
- 0 0 1 1 1 0 1

Find the Hamming distances between all pairs of codewords. Suppose that a codeword of $\Sigma$ is transmitted, and one, two or three bit errors occur; show that the received word is not a codeword (i.e., the error is detected).

What is $\textsf{DECODE}(0001110)$? Give an example of a word which has more than one nearest neighbour in $\Sigma$. Try to see if there are words with three, four etc. nearest neighbours. Try to write a word with a maximum possible number of nearest neighbours in $\Sigma$.

**Answer to E2.4.** The Hamming distances between all pairs of distinct codewords can be found directly by looking at the 28 possible pairs. (The number of pairs of distinct codewords is $M(M - 1)/2$.)

This can be optimised if one notices that the non-zero codewords are cyclic shifts of 1001110. Because applying a cyclic shift to both $v$ and $w$ does not change the Hamming distance between $v$ and $w$, it is enough to only find the distance $d(0000000, 1001110)$ and the distance from 1001110 to all the other codewords. All distances turn out to be 4. Hence $d(\Sigma) = 4$.

A 3-dimensional euclidean space analogue of this code is a set of four points with all pairwise distances equal to 4. Such points are vertices of a regular tetrahedron. In $n$-dimensional euclidean space, one can construct $n + 1$ points (not more) with equal pairwise distances. They are vertices of what is called an $n$-dimensional simplex. For this reason, the code $\Sigma$ which has a rare property that all pairwise distances between the codewords are the same, is called a **simplex code**.

Since $d(C) = 4$, the code detects up to $4 - 1 = 3$ errors and corrects up to $[(4 - 1)/2] = 1$ error.
The nearest neighbour of 0001110 in $\Sigma$ is 1001110 (by inspection), at distance 1. Hence $\text{DECODE}(0001110) = 1001110$.

Here is an example of a word with seven nearest neighbours in $\Sigma$: 1111111. All the non-zero codewords of $\Sigma$ are at distance 3 from this word. It looks as if there is no word equidistant from all codewords; check this!

**Remark.** More efficient ways of calculating the parameters of $\Sigma$ (and other simplex codes) will arise when we learn how to deal with linear codes, Hamming codes and cyclic codes. To give you a hint: in $\Sigma$, the *difference* of two codewords (viewed as vectors in $\mathbb{F}_2^7$) is again a codeword, so $d = 4$.

**Exercise 2.5.** Look again at the Manchester code, the Luhn code and the code $\Sigma$ introduced in the previous exercises. For each of these codes, determine the parameters $[n, k, d]$ of the code; state how many errors can the code detect and how many errors can the code correct.

**Answer to E2.5.** The Manchester code: $n = 2$, $q = 2$, $M = 2$ so $k = 1$; $d = 2$ by inspection. A $[2, 1, 2]_2$-code, can detect up to 1 bit error. Does not correct errors.

The Luhn code: $n = 16$, $q = 10$, $M = 10^{15}$ so $k = 15$, $d = 2$. A $[16, 15, 2]_{10}$-code. Can detect up to 1 symbol error. Does not correct errors.

The code $\Sigma$: $n = 7$, $q = 2$, $M = 8$ so $k = 3$; $d = 4$ by inspection. A $[7, 3, 4]_2$-code. Can detect up to 3 bit errors and can correct a single bit error.

**Exercise 2.6.** (a) Let $C$ be a $q$-ary code of length $n$. Denote $M = |C|$. Show: $n \geq \log_q M$.

(b) Assume that the cost of transmitting one symbol via a $q$-ary channel is $cq$. (*Imagine a $q$-ary channel as a cable with $q$ wires; the costs of building and maintaining it would be roughly proportional to $q$.) Suppose that you need to design a code with $M \gg 0$ codewords (that is: $M$ is a very large number), and you have the control over the length $n$ and the size $q$ of the alphabet. Which $q$ will ensure the lowest transmission costs per codeword? In particular, are the binary channels (the type most widely used in today’s computer networks) the most economical?

**Answer to E2.6.** (a) A code of length $n$ in an alphabet of size $q$ has at most $q^n$ codewords (an elementary counting argument), so $q^n \geq M$, hence the result.

(b) The cost of transmitting one codeword is $cq$. By part (a), $n \geq \log_q M$ so this cost is estimated from below as $cq \log_q M = K \frac{q}{\ln q}$ where the constant $K$ is $c \ln M$.

The function $f(x) = x/\ln x$ decreases on $(0, e)$ and increases on $(e, \infty)$ (*check this by differentiation or otherwise*) so $f(q) > f(3)$ if $q > 3$. Hence the only candidates for the
minimum are $q = 2$ and $q = 3$. Calculating $f(2) \approx 2.89$ and $f(3) \approx 2.73$, we conclude that — if we accept the (somewhat arbitrary) assumptions in the problem — ternary codes are the most economical.