Faculty of Mathematical Sciences

University of Twente

University for Technical and Social Sciences

P.O. Box 217
7500 AE Enschede
The Netherlands
Phone: +31-53-4893400
Fax: +31-53-4893114
Email: memo@math.utwente.nl

Memorandum No. 1588

Error patterns

C. Hoede and Z. Li[1]

September 2001

ISSN 0169-2690

[1]On leave from Shaanxi Normal University, Xi'an, P.R. China

Error Patterns

C. Hoede and Z. Li*

Faculty of Mathematical Sciences
University of Twente
P.O. Box 217
7500 AE Enschede, The Netherlands

Abstract

In coding theory the problem of decoding focuses on error vectors. In the simplest situation code words are $(0,1)$-vectors, as are the received messages and the error vectors. Comparison of a received word with the code words yields a set of error vectors. In deciding on the original code word, usually the one for which the error vector has minimum Hamming weight is chosen. In this note some remarks are made on the problem of the elements 1 in the error vector, that may enable unique decoding, in case two or more code words have the same Hamming distance to the received message word, thus turning error detection into error correction. The essentially new aspect is that code words, message words and error vectors are put in one-one correspondence with graphs.

Keywords: codes, graphs, error patterns.
2000 AMS Classification: 94B

1 Introduction

We quote one of the standard books on coding theory, that by Berlekamp [1], §1.1. "Suppose that we wish to transmit a sequence of binary digits across a noisy channel. If we send a one, a one will probably be received, if we send a zero, a zero will probably be received. Occasionally, however, the channel noise will cause a transmitted one to be mistakenly interpreted as a zero or a transmitted zero to be mistakenly interpreted as a one. Although we are unable to prevent the channel from causing such errors, we can reduce their undesirable effects with the use of coding. The basic idea is simple. We take a set of k message digits which we wish to transmit, annex to them r check digits, and transmit the entry block of $n = k + r$ channel digits. Assuming that the channel noise changes sufficiently few of these n transmitted channel digits, the r check digits may provide the receiver with sufficient information to enable him to defect and correct the channel errors.
We refer to the given reference for an excellent introduction to *code words* $\underline{c}$, together forming a *code*, that after sending give received *message words* $\underline{r}$ that

*On leave from Shaanxi Normal University, Xi'an, P.R. China

may show *errors* $\underline{e}$ with respect to the code word that was sent through the channel. With addition modulo 2 we have

$$\begin{aligned} \underline{r} &= \underline{c} + \underline{e} \quad \text{or} \\ \underline{c} &= \underline{r} + \underline{e} \quad \text{or} \\ \underline{e} &= \underline{r} + \underline{c} \quad . \end{aligned}$$

Usually the codes have an *information rate*, $R = k/n$, that may be low as the number r of check digits is taken to be large. Berlekamp states, "We are usually more interested in codes which have a higher information rate".
The "basic idea", as described by Berlekamp, is the concept of check digits. However, check digits are not absolutely necessary. Suppose we have a set of c code words $\underline{c}_1, \underline{c}_2, \ldots, \underline{c}_c$, that are $(0,1)$-vectors of, say, length n. Then in the set U of 2^n possible received messages $\underline{r}$, each vector has a certain *distance* to the code words. The usually chosen distance is the *Hamming distance* which simply is the number of elements one in the error vector $\underline{e} = \underline{r} + \underline{c}$. In U each vector $\underline{r}$ has a certain distance to each of the code words. Decoding then can take place by identifying the code word with minimum Hamming distance to $\underline{r}$. One may think of U as being partially partitioned into "spheres" around the code words. If $\underline{r}$ is within one of these spheres, the corresponding code word is chosen as the original. If $\underline{r}$ is outside these spheres, the distances to $\underline{c}_1, \ldots, \underline{c}_c$ can be calculated and the code word on shortest distance is chosen in the decoding procedure. In these cases one speaks of *error correction*. The interesting situation is that in which $\underline{r}$ has the same, shortest, distance to two or more code words, say $\underline{c}_1, \underline{c}_2$ and $\underline{c}_3$. Error correction is not possible in this case, one can only speak of *error detection*. Let the Hamming distance between $\underline{r}$ and the three code words $\underline{c}_1, \underline{c}_2$ and $\underline{c}_3$ be 7, and let the distance between $\underline{r}$ and other code words be larger. Than we say that an error of weight 7 has been detected, and that no error correction can take place.
Let us look at this situation in terms of the three error vectors $\underline{e}_1 = \underline{r} + \underline{c}_1$, $\underline{e}_2 = \underline{r} + \underline{c}_2$ and $\underline{e}_3 = \underline{r} + \underline{c}_3$. These are three $(0,1)$-vectors with 7 elements, out of n, being 1.
It looks as if there is little we can do to turn detection of errors into correction of errors, i.e. determining which of the three code vectors $\underline{c}_1, \underline{c}_2$ and $\underline{c}_3$ is the most likely original of $\underline{r}$. In this note it is studied how the *pattern* of the ones in the error vector can be used to decide between the three competing code vectors. We will discuss several patterns and the way they can be used or interpreted.

2 Burst errors

Let $n = 20$ and let the three error vectors of our example in Section 1 look like

$$\begin{aligned} \underline{e}_1 &= (0,1,0,0,1,0,1,0,0,0,1,0,0,1,0,1,0,1,0,0) \\ \underline{e}_2 &= (0,0,0,1,0,1,0,0,0,1,1,1,1,0,0,0,1,0,0,0) \\ \underline{e}_3 &= (0,0,0,0,1,1,1,0,0,0,0,0,0,1,1,1,1,0,0,0). \end{aligned}$$

There is a difference in the patterns of these error vectors. $\underline{e}_1$ seems a rather random occurrence of errors, $\underline{e}_2$ shows a *burst* of 4 consecutive errors and $\underline{e}_3$ shows two bursts, of length 3 and length 4, and no other errors.
One way to take the pattern into account is to define a burst as a set of consecutive errors of cardinality above a certain integer. Such an integer might be chosen on probabilistic grounds by calculating the probability of a burst of length b and set some bound on this probability, e.g. 0.01. Whenever b consecutive errors occur with probability smaller than 0.01, they are considered to form a burst.
Suppose for our example error vectors, we talk of a burst in case there are 3 or more consecutive errors. Then these may be considered to be special errors that can be dealt with in a special way, by *burst reduction*.
By this is meant that the errors of the bursts are removed from the error vectors that now have 7, 3 respectively 0 ones. In this case the third error vector $\underline{e}_3$ hints at $\underline{c}_3$ being the original code word. We have reduced as it were the distance between $\underline{r}$ and $\underline{c}_3$, as it appeared from the original error vector, from 7 to 0.
Suppose now that $b = 3$ is not considered to indicate a burst but $b = 4$ is. Then the burst reduction only removes the bursts of length 4 and the resulting reduced error vectors $\underline{e}_1^r, \underline{e}_2^r$ and $\underline{e}_3^r$ have 7, 3 respectively 3 ones. Whereas in the former case error detection was turned into error correction, here we still have only error detection.

This analysis can be extended in the following way where, for the ease of calculation, we assume *cyclic conditions* on the code words and error vectors. This means that last and first digit are considered to be consecutive. The zeroes and ones can then be seen as positioned on a circle. n can be seen as the "length" of this circle, that is divided by the k ones into k parts of lengths $d_1, d_2, \ldots, d_k$, with $d_1 + d_2 + \ldots + d_k = n$. For our three example error vectors we have $n = 20$, $k = 7$ and the distances, in vector representation, are

$$\begin{aligned} \underline{d}_1 &= (3, 2, 4, 3, 2, 2, 4) \\ \underline{d}_2 &= (2, 4, 1, 1, 1, 4, 7) \\ \underline{d}_3 &= (1, 1, 7, 1, 1, 1, 8). \end{aligned}$$

We agreed that $\underline{e}_3$ was "burstier" that $\underline{e}_2$ and $\underline{e}_1$ and that $\underline{e}_2$ was "burstier" than $\underline{e}_1$. The *burstiness* B of an error vector might be used to obtain error correction, next to the Hamming distance. Hence we want to develop a measure for B.
Now, intuitively, burstiness means many consecutive ones, in the extreme case all ones being consecutive in the error vector, creating one burst. Low burstiness means that the occurring errors are not related and, intuitively, are *distributed* over the circle. We might therefore also consider a measure for the *distribution* D. High distribution means low burstiness and low distribution means high burstiness.

We focus on the distribution measure D.

Let us multiply the k distances to obtain $P_k = \prod_{i=1}^{k} d_i$. The maximum value for P_k occurs when the circle is divided by the k ones into equal parts of length $\frac{n}{k}$.

Compare this with the physical situation of k electric charges, say electrons, on a circular wire. They repel each other and will distribute in the described way over the circle. But the statement is also easily proven mathematically. We admit non-integral values for the distances in this proof.

Lemma 1. P_k has maximum value $\left(\frac{n}{k}\right)^k$.

Proof. Consider an interval of length a and a point on it so that the two parts of the interval have lengths x and $a-x$. The product $x(a-x)$ is maximum for $x = \frac{a}{2}$.
Now consider an arbitrary distribution of the k ones. Whenever two consecutive intervals are different in length, the distribution can be changed into one in which the intervals are equal, by shifting the one that both intervals have in common to the middle of the joint interval. The value of P_k for that distribution will be larger.
The maximum for P_k is obtained for that distribution in which all k intervals have the same length $\frac{n}{k}$. □

Although $\frac{n}{k}$ need not be integer, we take the value $(\frac{n}{k})^k$ to norm the product P_k and obtain

Definition 2. The *distribution* of an error vector of length n with k ones is

$$\begin{aligned} D &= \frac{\prod_{i=1}^{k} d_k}{(\frac{n}{k})^k} = \frac{\prod_{i=1}^{k} d_k \cdot k^k}{n^k} \qquad (k \geq 1) \\ D &= 1 \qquad (k = 0) \quad . \end{aligned}$$

Clearly we have $0 \leq D \leq 1$. In case there are no errors, $\underline{e} = \underline{0}$, we have defined D to be 1 as we also define burstiness now as follows.

Definition 3. The *burstiness* of an error vector $\underline{e}$ of length n with k ones is

$$B = 1 - D,$$

where D is the distribution of $\underline{e}$ given in Definition 2.

Clearly if $k = 0$ we would like to have the error vector $\underline{e}$ to have burstiness 0.

Let us see whether these measures make sense. For $k = 1$ we have one interval, due to the cyclicity, of length n. We obtain $D = \frac{n \cdot 1^1}{n^1} = 1$ and $B = 0$. One error is not considered a burst!
For our three example error vectors we have $(\frac{20}{7})^7$ in all three cases as the denominator of D and respectively $2^3 \cdot 3^2 \cdot 4^2$, $1^3 \cdot 1^1 \cdot 4^2 \cdot 7^1$ and $1^5 \cdot 7^1 \cdot 8^1$ in the denumerator of D, leading to values $0.74, 0.24$ and 0.06 for the distribution D and $0.26, 0.76$ and 0.94 for the burstiness B of $\underline{e}_1, \underline{e}_2$ and $\underline{e}_3$.

If one accepts these measures, one should be aware of the following. It may be that a vector without consecutive ones has higher burstiness than one that does have two consecutive ones. Compare the vectors

$$\begin{aligned} \underline{e}_4 &= (1\,0\,1\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0) \quad \text{and} \\ \underline{e}_5 &= (1\,1\,0\,0\,0\,1\,0\,0\,0\,0\,1\,0\,0\,0\,0), \end{aligned}$$

with $n = 15$ and $k = 4$.
The first vector $\underline{e}_4$ has no burst but the four ones are very close, whereas the second vector $\underline{e}_5$ has a burst of length 2 but the rest of the ones is well distributed. Which vector is burstier? $\underline{e}_4$ has $2^3 \cdot 9^1 = 72$ in the denumerator, whereas $\underline{e}_5$ has $1^1 \cdot 4^1 \cdot 5^2 = 100$ in the denumerator. $(\frac{n}{k})^k = (\frac{15}{4})^4 \approx 198$, so that we find about 0.36 and 0.50 for D, so 0.64 and 0.50 for B respectively. In our opinion $\underline{e}_4$ should be considered "burstier" than $\underline{e}_5$, in spite of the fact that no two ones are consecutive.
Now we have a measure, next to the Hamming distance, that may enable us to turn error detection into error correction. In order to see this happen we consider a code consisting of the following four code words of length 6.

$$\begin{aligned} \underline{c}_0 &= (0,0,0,0,0,0) \\ \underline{c}_1 &= (1,1,1,1,0,0) \\ \underline{c}_2 &= (1,0,1,0,1,1) \\ \underline{c}_3 &= (0,1,0,1,1,1) \quad . \end{aligned}$$

There are 64 possible received words $\underline{r}$. The four code words have, pairwise, distance 4. 28 vectors $\underline{r}$ have Hamming distance 1 to one of the code words and can therefore be corrected. The 36 vectors that show two or more error vectors with the same minimum Hamming weight, so that we cannot correct to one of the four code words, exhibit 2,3,4 or 6 ones.

Now we try to obtain correction by applying the measure B and assuming that if an error vector is burstier, then the corresponding code word is more likely to have been sent. Consider, for example, the received word (0,0,0,0,1,1), that has error vectors

$$\begin{aligned} \underline{e}_0 &= (0,0,0,0,1,1) \\ \underline{e}_1 &= (1,1,1,1,1,1) \\ \underline{e}_2 &= (1,0,1,0,0,0) \\ \underline{e}_3 &= (0,1,0,1,0,0) \end{aligned}$$

with respect to the four code words. $\underline{e}_1$ has Hamming weight 6, the other three vectors have Hamming weight 2. So there are three candidates for the decoding. The product P_k is 5, 8 and 8 for these three candidate error vectors respectively. So D is smallest for $\underline{e}_0$ or, equivalently B is largest. So we have now $\underline{c}_0$ as the most likely candidate for our decoding.

The only received words that cannot be decoded still are, with the error vectors,

$$\begin{array}{lllllllllll} \underline{r} & = & (1,1,0,0,0,0) & \underline{e}_0 & = & (1,1,0,0,0,0) & \underline{e}_1 & = & (0,0,1,1,0,0) \\ \underline{r} & = & (0,0,1,1,0,0) & \underline{e}_0 & = & (0,0,1,1,0,0) & \underline{e}_1 & = & (1,1,0,0,0,0) \\ \underline{r} & = & (1,0,0,1,1,1) & \underline{e}_2 & = & (0,0,1,1,0,0) & \underline{e}_3 & = & (1,1,0,0,0,0) \\ \underline{r} & = & (0,1,1,0,1,1) & \underline{e}_2 & = & (1,1,0,0,0,0) & \underline{e}_3 & = & (0,0,1,1,0,0) \end{array} .$$

So in these four out of 64 cases neither the Hamming distance nor the burstiness leads to identifying a most likely code word sent. Our effort will be focussed on finding a way to make a further distinction for a situation like this.

Burst patterns like we considered here were only chosen as an example to show how an extra consideration, burstiness next to the Hamming weight of an error vector, may turn error detection into error correction, *i.e.*, unique decoding. In the next section we give a more sophisticated interpretation of error patterns.

3 Graph patterns

We consider complete graphs first and refer to Bondy and Murty [3] for graph terminology. In particular we are interested in K_{2n+1}, $n \in \mathbb{N}$, so the complete graph with an odd number of vertices, having even degree $2n$. This graph has many eulerian tours. When the vertices are labeled $0, 1, 2, \ldots, 2n$ the edges can be partitioned into n classes by considering the differences of the labels modulo $2n+1$. If i and j are two labels either $i-j$ or $j-i$ modulo $2n+1$ is an integer t in $[1, n]$. The edge with vertices with labels i and j is said to be of *type* t. It was proven by Jetten [7], that a *canonical* eulerian tour exists, that starts in an arbitrary vertex with an edge of type 1, followed by edges of type $2, 3, \ldots, n$, whereafter the sequence of edges of types 1 up to n is repeated.
K_{2n+1} has $\binom{2n+1}{2} = n(2n+1)$ edges, exactly $2n+1$ of each type $1, 2, \ldots, n$. As an example in Figure 1 K_7 is considered. The canonical eulerian tour has edges of type 1, 2 and 3, and is indicated by arrows.

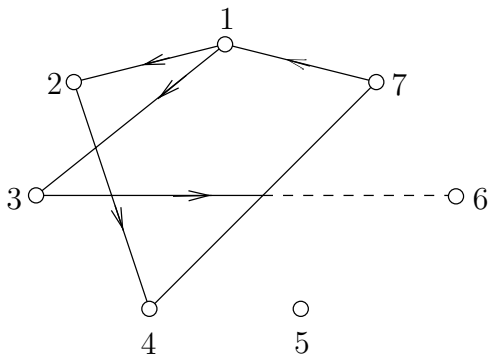

Figure 1: K_7 with first six edges of a canonical Euler tour $1 \to 2 \to 4 \to 7 \to 1 \to 3 \to 6 \to \ldots$ starting from vertex 1.

As there are $\binom{7}{2} = 21$ edges we can associate a $(0,1)$-vector of length 21 with each subgraph G of K_7, by choosing representation 1 if the edge corresponding

to the element of the vector is present and 0 otherwise. The order of the elements in the vector may be chosen according to an eulerian tour of K_7, in particular according to the canonical eulerian tour. It is evident that the following lemma holds.

Lemma 4. There is a one to one correspondence between the $(0,1)$-vectors of length $n(2n+1)$ and the subgraphs of K_{2n+1}.

The importance of this simple lemma is that code word $\underline{c}$, message word $\underline{r}$ and error vector $\underline{e}$ all three show a certain pattern, namely the graph that corresponds to them!
This means that a code can be seen as a set of graphs, subgraphs of some complete graph in the situation considered sofar, and that the way of looking at $\underline{c}$, $\underline{r}$ and $\underline{e}$ has changed. If we set K_{2n+1} as a picture on a TV-screen, and every subgraph as well, then $\underline{c}$ is some graph that is received as some distorted graph $\underline{r}$, the distortion also being a graph $\underline{e}$. Suppose now that just the $(0,1)$-vectors are considered and that no correction does take place as there are two or more $(0,1)$-vectors $\underline{c}$ with the same Hamming distance to the received word $\underline{r}$, then the code words, as graphs, may give a clue to the error correction. To make this idea clear we consider four code words in the form of four pictures and assume they correspond to certain $(0,1)$-vectors. The received words also have the form of a picture with corresponding $(0,1)$-vector, see Figure 2.

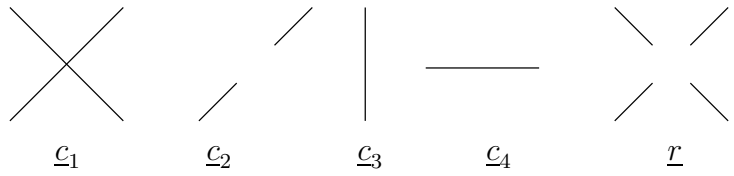

Figure 2: Code words in the form of pictures.

Just for the sake of illustration, let there be $(0,1)$-vectors for $\underline{c}_1, \underline{c}_2, \underline{c}_3, \underline{c}_4$ and $\underline{r}$ with the property that the Hamming distance between $\underline{r}$ and $\underline{c}_1$ and between $\underline{r}$ and $\underline{c}_3$ are the same and that this distance is smaller than that between $\underline{r}$ and the other two code words $\underline{c}_2$ and $\underline{c}_4$. Then the candidates for decoding are $\underline{c}_1$ and $\underline{c}_3$ as far as the Hamming distance is concerned. However, looking at the pictures of $\underline{c}_1, \underline{c}_2$ and $\underline{r}$, we immediately notice the *similarity* between $\underline{r}$ and $\underline{c}_1$ and want to identify $\underline{c}_1$ as the original message.
What happens here is that the *graph pattern* is used as an extra check to turn the error detection into an error correction. Looking on code words as graphs ties up decoding with *pattern recognition*, providing an extra check for the decoding procedure. Of the many techniques in that field we only mention the use of *similarity measures*. For graphs these have been developed by Hoede [6] in the context of knowledge representation. Having an appropriate similarity measure $SIM(G_1, G_2)$ for the similarity of two graphs G_1 and G_2, decoding then consists of two steps.

1. The Hamming distance between code words $\underline{c}$ and received word $\underline{r}$ are calculated.

2. In case no unique decoding can take place, as more code words have the same distance to $\underline{r}$, the similarity between the graphs corresponding to these code words on one side and the graph corresponding to $\underline{r}$ are calculated. The code words with the graph showing the highest similarity is chosen.

Some remarks are due here. First, it may happen that even the similarities turn out to be the same. Second, in principle it is not necessary to carry out the first step at all. Once code words are set into one-one correspondence to graphs, similarity calculation may suffice. Third, if error correction still did not take place some third criterium might be developed to achieve this.

A drawback of the presented use of graph patterns is that we consider complete graphs K_{2n+1}, so that the length of the code words is $3, 10, 21, 36, \ldots, \binom{2n+1}{2}$, etc. However, any graph that allows a eulerian tour gives a one-one correspondence between graphs and code words satisfying the cyclicity condition (without this condition any graph with the appropriate number of edges may be chosen). An interesting example was developed in relation to the Ising problem in statistical physics, see Hoede [5].
Consider a $n \times (n+1)$ quadratic lattice with toroidal boundary conditions. All vertices have degree 4 and there are two types of edges, horizontal and vertical edges. Due to the choice of the dimensions, n in vertical and $n+1$ in horizontal direction, there is a canonical eulerian tour in which alternatingly a horizontal and a vertical edge are chosen, see Figure 3 for the 3×4 lattice.

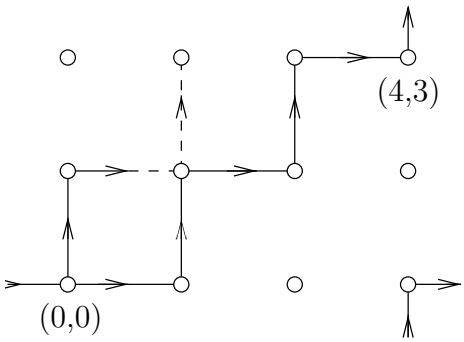

Figure 3: 3×4 quadratic lattice with canonical eulerian tour. Starting in vertex $(0,0)$ eight edges have been drawn solidly.

The $(0,1)$-vectors have length $2n(n+1)$ now and correspond to subgraphs of the quadratic lattice. So here we have a way to represent code words of length $4, 12, 24, 40, \ldots$, etc. Thinking of the graphs as pictures on a TV-screen again, the fact that TV-pictures are built up from pixels on a square lattice makes clear that virtually any picture may be represented by a $(0,1)$-vector corresponding to a subgraph of a square lattice like we considered here.

We have assumed a cyclicity condition on the code and certain structures of the graph, that caused a restriction on the length of the code words. We might

of course have taken a simple cycle of length n when considering code words of length n, but then the graph structure does not give more information than the cyclic array of zeroes and ones of the code words themselves. We would not go beyond what was discussed in Section 2.

Sofar we only considered subgraphs of unlabeled graphs as code words. Edges were present or not, represented by a one or a zero respectively. We might of course also have used two colors, white and black or any other two colors to get an edge-colored subgraph, of K_{2n+1} say, as corresponding to a $(0,1)$-vector. The TV-screen needed was a simple black and white screen. Now let us allow that there are more than two colors for the edges. We would now need a color TV-screen to represent our code words, that now have elements chosen from a set of three values, say $\{-1,0,+1\}$. The ideas presented easily generalize to this situation. The first step, comparison of $(-1,0,+1)$-vectors, yields error correction or error detection, the second step may be used to turn detection into correction by calculating the similarity of graphs that are edge-colored. There is, in principle, no limit to the number of colors that may be used. There may be an infinite scale of colors, like in the rainbow. This brings us to an interesting generalization that is investigated with great interest recently.

4 Quantum patterns

An infinite number of colors could be encoded by the complex numbers $e^{i\varphi}$, $\varphi \in [0, 2\pi]$, or just by φ. The code vectors now correspond to multicolored graphs. The numbers correspond to vectors of length 1 in the two-dimensional space of complex numbers. Such a space may be associated to every element of a code vector, that are therewith elements of the tensor product of n of such spaces.

The interesting thing is that code vectors like these are currently investigated in *quantum coding theory*, in which the errors occurring are shifts in the phase φ of the elements and there is interrelation between the various elements, called *entanglement*, very much the way there are patterns in the cases we considered sofar. We will now investigate this relationship further.

We recall some of the definitions given by Calderbank and Shor [2], who showed in 1996 that good quantum error-correcting codes exist.
The definition of their quantum codes relies heavily on that of normal linear codes. The basis of the description of such a linear code is the field F_2, of two elements 0 and 1. A *linear* code is a subspace of F_2^n (the n-dimensional vector space over the field F_2). Socalled *quantum* codes are described after introducing a *quantum Hilbert space*. The simplest such space is $\mathcal{H}_2$ which is the *complex* space generated by basis vectors $|b_0\rangle$ and $|b_1\rangle$, where we use the socalled bra- and ket-vector notation that is usual in quantumphysics. The state of some physical operator is described by a ket-vector $|\,\rangle$. The bra-vector $\langle\,|$ also denotes a state, the terminology being due to the fact that the inner product of two vectors can now be described by the bracket(s) $\langle a|b\rangle$ for the vectors $\langle a|$ and $|b\rangle$. Note that in linear algebra the inner product of the vectors

$\underline{a}$ and $\underline{b}$ is usually seen as the matrix multiplication of a $(1 \times n)$-matrix with a $(n \times 1)$-matrix.

$\mathcal{H}_2$ is called a *qubit*. The complex space $\mathcal{H}_2^n$ over n qubits is the space generated by basisvectors $|b_0\rangle, |b_1\rangle, \ldots, |b_2 n_{-1}\rangle$ where b_i is the representation of the number i in binary. So for $n = 3$ we have the eight vectors $|000\rangle, |001\rangle, \ldots, |111\rangle$. This Hilbert space has a natural representation as a tensor product of n copies of $\mathcal{H}_2$, with the i-th copy corresponding to the i-th bit of the basisvectors.
Now we recall that $\mathcal{H}_2^n$ is a *complex* space where, in general, the j-th bit of the vector is some complex number $e^{i\varphi_j}$. φ_j is called the *phase* of the j-th bit. The general vector looks like $(e^{i\varphi_1}, e^{i\varphi_2}, \ldots, e^{\varphi_n})$, which is the encoding for the coloring with an infinite number of colors that we discussed at the end of the former section.

Calderbank and Shor used the $[7, 4, 3]$ Hamming code as an example to illustrate the construction of quantum error-correcting codes. This code has code words of length 7, has $2^4 = 16$ code vectors, so dimension $d = 4$, and correct one error as the minimum distance between code words in 3. The code vectors are

$\underline{c}_1$	0000000	$\underline{c}_9$	1111111
$\underline{c}_2$	1001110	$\underline{c}_{10}$	1011000
$\underline{c}_3$	0100111	$\underline{c}_{11}$	0101100
$\underline{c}_4$	1010011	$\underline{c}_{12}$	0010110
$\underline{c}_5$	1101001	$\underline{c}_{13}$	0001011
$\underline{c}_6$	1110100	$\underline{c}_{14}$	1000101
$\underline{c}_7$	0111010	$\underline{c}_{15}$	1100010
$\underline{c}_8$	0011101	$\underline{c}_{16}$	0110001.

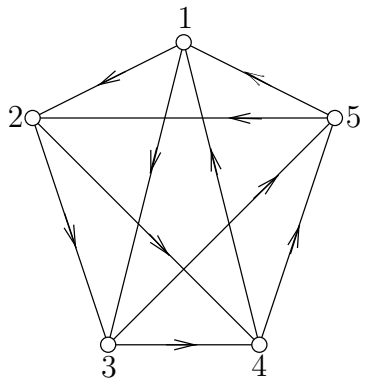

Figure 4: Canonical eulerian tour in K_5: $1 \to 2 \to 4 \to 5 \to 2 \to 3 \to 5 \to 1 \to 3 \to 4 \to 1$.

The first half of the 16 vectors has even weight and are orthogonal to all 16 vectors. These 8 vectors of the code C_1 form the dual code $C_1^\perp$. The construction of quantum states, there turn out to be only two, $|c_0\rangle$ and $|c_1\rangle$, relies on the structure of these code in the sense that $C_1^\perp$ is a code contained in the larger code C_1. We want to show that this structure can also be interpreted in

a graph theoretical way. This then means that the process of quantum coding and decoding can also be interpreted in a graph theoretical way.

We recall the one-one correspondence between code vectors and subgraphs of the complete graph with odd number of vertices. As we have code words of length 7 and we want to embed in a larger quantum Hilbert space we choose $n = 10$ and consider the graph K_5. In terms of Calderbank and Shor we consider a quantum error-correcting code as a unitary mapping of $\mathcal{H}_2^7$ into $\mathcal{H}_2^{10}$.

The canonical eulerian tour of K_5 has alternatingly edges of type 1 and type 2, see Figure 4.

Each vertex of K_5 has two image vertices on the tour, as each vertex has degree 4. Consider vertex 1 and draw the tour as in Figure 5.

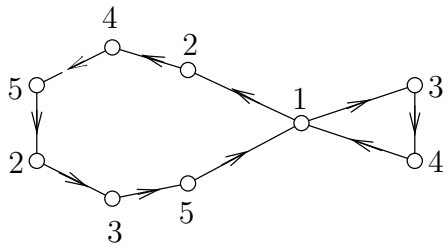

Figure 5: Alternative drawing of the eulerian tour.

In Figure 5 we see that the tour has one part of length 3 and one part of length 7. We now let the bits of the code words correspond to the edges of the part of length 7. In this way each code vector determines, by the bits that are one, a set of edges in K_5. In Figure 6 the subgraphs corresponding to the 16 code words are given.
One sees in Figure 6 that the graphs corresponding to the code words $\underline{c}_1, \underline{c}_2, \ldots, \underline{c}_8$, have an even number of edges in common with the other 15 graphs, due to the orthogonality of $C_1^{\perp}$. Let $\mathcal{H}_{C_1}$ be the subspace of $\mathcal{H}_2^{10}$ generated by vectors $|c\rangle$ with $c \in C_1$. Let M be a generator matrix for C_1; this means that C_1 is the row space of M, so that vM ranges over all the code words in C_1 as v ranges over all vectors in $F_2^{\dim(C_1)}$. In our example $\dim(C_1) = 4$, so there are $2^4 = 16$ vectors v. Calderbank and Shor now define, for $w \in F_2^n$, a *quantum state* $|c_w\rangle$ by

$$|c_w\rangle = 2^{-\dim(C_1)/2} \sum_{v \in F_2^{\dim(C_1)}} (-1)^{vMw} |vM\rangle.$$

Based on the example Hamming code a *quantum code* is constructed with only two code words $|c_0\rangle$ and $|c_1\rangle$ where

$$\begin{aligned} |c_0\rangle &= \frac{1}{4}(|\underline{c}_1\rangle + \ldots\ldots\ldots\ldots\ldots\ldots + |\underline{c}_{16}\rangle) \\ |c_1\rangle &= \frac{1}{4}(|\underline{c}_1\rangle + \ldots + |\underline{c}_8\rangle - |\underline{c}_9\rangle - \ldots - |\underline{c}_{16}\rangle). \end{aligned}$$

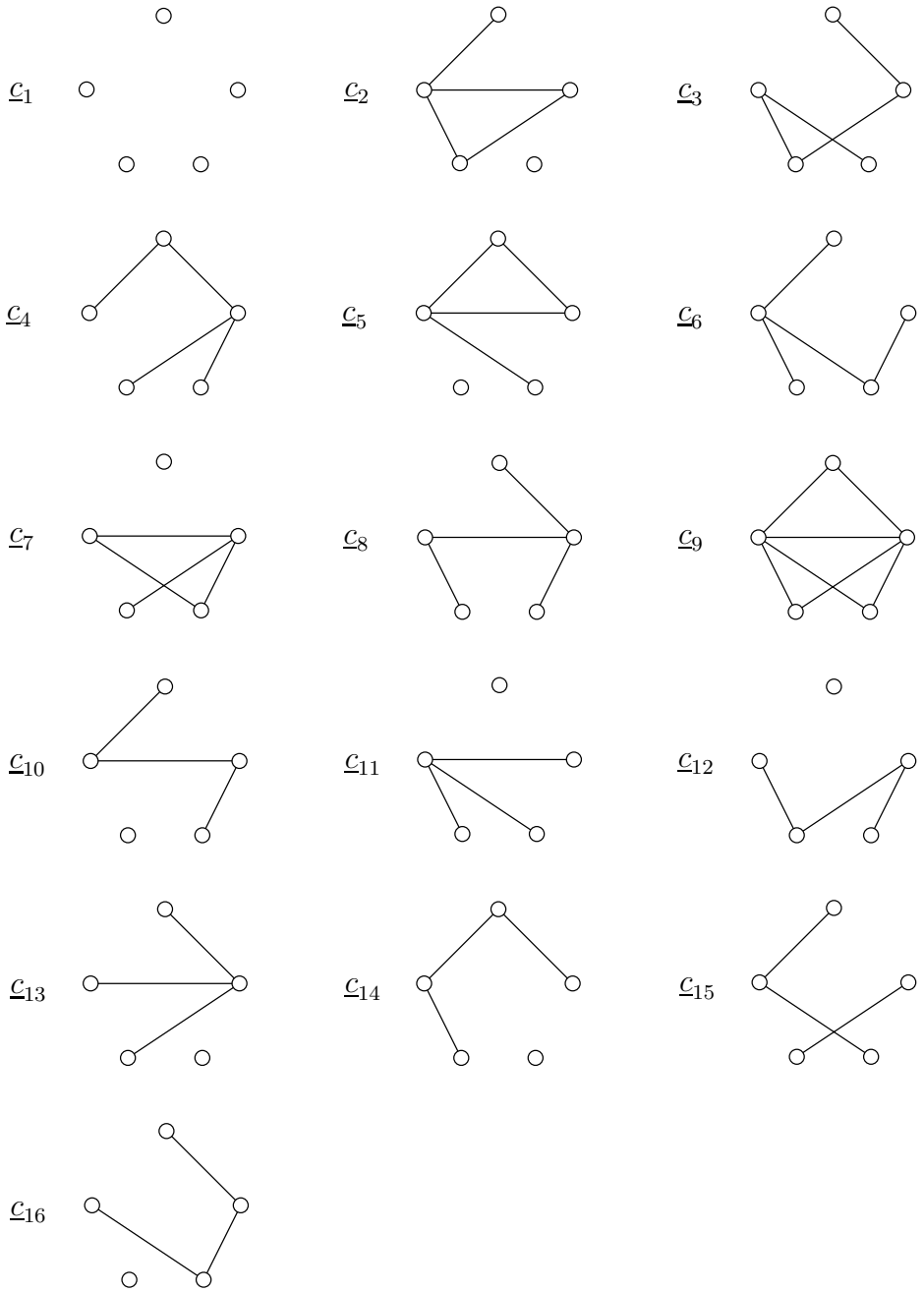

Figure 6: Graphs corresponding to code words.

Trying to interpret these quantum code words in terms of the 16 graphs in Figure 6, we may look upon the quantum states as states of the graph K_5.
With each edge corresponds a qubit $\mathcal{H}_2$ that has two basis vectors $|0\rangle$ and $|1\rangle$. These two basic quantum states may be identified with the real and the imaginary unit, 1 and i, of the complex plane. The normalized states, length 1, in $\mathcal{H}_2$ are then $|\cos\varphi\cdot 1+\sin\varphi\cdot i\rangle = |e^{i\varphi}\rangle$, $\varphi\in[0,2\pi]$. The two basis vectors can therefore also be represented as $|0\rangle = |e^{i\cdot\frac{\pi}{2}}\rangle$ and $|1\rangle = |e^{i\cdot 0}\rangle$. The "state" of an edge is, in general, $|e^{i\varphi}\rangle$ for some $\varphi\in[0,2\pi]$. The states $|\underline{c}_1\rangle,\ldots,|\underline{c}_{16}\rangle$ can therefore been seen as special states of K_5, the general state being $|e^{i\varphi_1},e^{i\varphi_2},\ldots,e^{i\varphi_{10}}\rangle$. Calderbank and Shor define a *quantum error-correcting code* Q with rate k/n as a unitary mapping of $\mathcal{H}_2^k\otimes\mathcal{H}_2^{n-k}$ into $\mathcal{H}_2^n$, where the quantum state in $\mathcal{H}_2^{n-k}$ is taken to be that where all the qubits have quantum state $|0\rangle$. For our example three qubits are taken to be in this state. The graphs corresponding to the code vectors $\underline{c}_i$ have edges that are in state $|1\rangle$ and nonedges that are in state $|0\rangle$. If we take φ in $|e^{i\varphi}\rangle$ to be the color of the edge, corresponding to this qubit state, then in all 16 cases the K_{10} is edge-colored with colors 0 (for edges in state $|1\rangle$) and $\frac{\pi}{2}$ (for edges in state $|0\rangle$).
The code words $|c_0\rangle$ and $|c_1\rangle$ of the constructed quantum code are superpositions of states $|\underline{c}_1\rangle$, $i=1,\ldots,16$, that were interpretable as colored graphs K_5. The interpretation of these *entangled* states is not very clear. The coefficients of the states are $+1$ or -1. These may be seen as multiplication of the elements with $e^{i\cdot 0}=1$ respectively $e^{i\cdot\pi}=-1$, or as non-shifting respectively shifting the colors of the edges by π. The superpositions of states $|\underline{c}_i\rangle$ in $|c_0\rangle$ then only differ in the sense that two different sets of colorings of 16 graphs K_5 are considered. A change of basis to each of the bits of a code word $|c_w\rangle$,

$$\begin{aligned} |0\rangle &\rightarrow \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle) \\ |1\rangle &\rightarrow \frac{1}{\sqrt{2}}(|0\rangle-|1\rangle), \end{aligned}$$

in the interpretation of colored edges, now is interpreted as a change from colors 0 respectively $\frac{\pi}{2}$ to colors $\frac{3\pi}{4}$ respectively $\frac{\pi}{4}$. For the example quantum code the *rotated* basis leads to states

$$\begin{aligned} |s_0\rangle &= \frac{1}{2\sqrt{2}}(|\underline{c}_1\rangle+|\underline{c}_2\rangle+\ldots+|\underline{c}_8\rangle) \quad \text{and} \\ |s_1\rangle &= \frac{1}{2\sqrt{2}}(|\underline{c}_9\rangle+|\underline{c}_{10}\rangle+\ldots+|\underline{c}_{16}\rangle). \end{aligned}$$

Cancellation of states, $|a\rangle-|a\rangle=0$, can be interpreted as cancellation of graphs with colorings that differ precisely π for every edge (qubit). The change of basis causes the superposition of two times 16 colored graphs to change into a superposition of two times 8 colored graphs.

A final remark should me made on the concept of *decoherence*. An original state $|x\rangle\in\mathcal{H}_2^k$ is encoded by Q to an encoded state $Q\mid x\rangle$, which, on transmission,

can be decohered, modeled by an arbitrary unitary transformation D on some of the qubits, and leading to a decohered encoded state $DQ \mid x\rangle$. The decoding problem consists of recovering the original state $|x\rangle$. Here we just remark that the decoherence in some of the qubit states can be interpreted as some color shift of the corresponding edges of a graph, in our example the graph K_5.
It is now clear that also in the case of quantum errors, color shifts of edges, the structure of the graph, the quantum error pattern might help in the decoding process. This idea is to be investigated further.

5 Codes, graphs and patterns

The idea behind error patterns is that the decoding process may be helped by extra information. Such extra information is present in graphs that were put in one to one correspondence with code vectors. The similarity of graphs, a subject still to be discussed, may turn error detection into error correction. In a way similarity defines a distance on top of the "normal" distance between code words as given e.g. by the Hamming distance. If $SIM(G_1, G_2)$ is some similarity measure, with

$$0 \leq SIM(G_1, G_2) \leq 1,$$

then a "distance" between G_1 and G_2 may be defined as $DIST(G_1, G_2) = 1 - SIM(G_1, G_2)$. For this concept of distance we also would have

$$0 \leq DIST(G_1, G_2) \leq 1.$$

A graph, when drawn in the plane, can be seen as a *pattern*. As there are no prescriptions for drawing a graph, the pattern contains more information than the graph. The specific way of drawing the graph introduces the distances in the plane of drawing, of the vertices of the graph. The situation is therefore the following.
A set $\mathcal{C}$ of code words can be represented by a set $\mathcal{G}$ of graphs, that can be represented by a set $\mathcal{P}$ of patterns. Graphs contain more information than code words and patterns contain more information than graphs. This hints at an important role that *pattern analysis*, see Grenander [4], can play in connection with the decoding problem. We will shortly consider six problems, mentioned by Grenander and will try to formulate analogous problems for graphs and codes.

Code words $\underline{c}$ give rise to received message words $\underline{c}^D$, the vectors that we have indicated by $\underline{r}$ in the former sections. Graphs G give rise to *distorted* graphs G^D, and patterns P give rise to *deformed* pattern P^D. The six problems considered by Grenander are

a) Image restoration

b) Image analysis

c) Image approximation

d) Pattern recognition

e) Image description

f) Patterns inference and abduction.

We will now try to formulate these problems for graphs and codes as well. As the pattern is "richer" than the graph or the code we may expect that not all six problems have a natural counterpart.

Problem a (image restoration)

This is the problem of finding a mapping from $\mathcal{P}^D$ to $\mathcal{P}$, $\mathcal{G}^D$ to $\mathcal{G}$ respectively $\mathcal{C}^D$ to $\mathcal{C}$. The mapping is supposed to restore the pure P, G respectively $\underline{c}$ which were deformed by D into the observed P^D, G^D, respectively $\underline{c}^D$.

This problem is the basic decoding problem in case of codes and can be formulated in all three cases.

Problem b (image analysis)
"Given an image I find a configuration c that gives rise to I. This involves finding the generators and combinatory relations of c."

From this quotation we see that patterns are seen as built up from generators and combinatory relations. A similar statement can be made for graphs, seeing the vertices as "generators" and the edges as "combinatory relations". For code words the digits may be seen as "generators", but the "combinatory relations" are unclear. It is precisely for this reason that we considered graph error patterns in Section 3.

Problem c (image approximation)
Let $\mathcal{P}^*$ be an additional pattern set with $\mathcal{P} \subset \mathcal{P}^*$, we want to find a "good" mapping $\mathcal{P}^D \to \mathcal{P}^*$ such that the P^* is in some sense close to $P \in \mathcal{P}$.

This problem can also be posed for graphs and code words. The main aspect is, of course, the concept of a "good" mapping. This asks for some distance concept, e.g. the Hamming distance for codes, which for graphs might be based on the concept of similarity.

Problem d (pattern recognition)
Give P^D find the pattern class $\mathcal{P}_r$ to which P belongs.

The new aspect here, with respect to problem a, is the concept of *pattern class*. Analogously, we can think of *graph class* and *code word class* for an analogous setting.

Problem e (image description)
Given $\mathcal{P}$ and $\mathcal{P}_*$ find a mapping $\mathcal{P} \to \mathcal{P}_*$ such that the $P_* \in \mathcal{P}_*$ is a "good" representative of P.

Even more than in Problem b this problem leans heavily on the fact that patterns contain more information than graphs or codes, for which no obvious analogous problem can be indicated.

Problem f (pattern inference and abduction)

"Given elements from $\mathcal{P}^D$ make inferences concerning $\mathcal{P}$ and the underlying regularity structure".

This problem is so general in nature that on one hand one can pose the problem for graphs and code words as well, but on the other hand we may quote Grenander: "At this level of generality, the question is almost meaningless, and it is only when we get to more detailed and concrete pattern (graph, code word) systems that we can start to offer specific methods of solution".

It will be our goal to study the possibility of exploiting the differences in information contained in patterns, graphs and code words, for some of these problems, focusing first on explicit examples.

References

[1] Berlekamp, E.R., *Algebraic Coding Theory*, McGraw-Hill, New York (1968).

[2] Calderbank, A.R. and P.W. Shor, Good quantum error-correcting codes exist, *Phys. Rev. A*, 54: 1098 (1996).

[3] , Bondy, J.A. and U.S.R. Murty, *Graph Theory with Applications*, MacMillan, London (1976).

[4] Grenander, U., *Pattern Analysis*, Springer-Verlag, New York (1978).

[5] Hoede, C., *On the Ising Problem*, Schroedersche Buchdruckerei, Diepholz (1968).

[6] Hoede, C., *Similarity in Knowledge Graphs*, Memorandum nr. 550, Faculty of Mathematical Sciences, University of Twente (1986).

[7] Jetten, A., Private communication. See C. Hoede, *On Canonical Euler Cycles*, Memorandum nr. 7, Faculty of Mathematical Sciences, University of Twente (1970).