A Viterbi decoder

Citation for published version (APA):

Document status and date:
Published: 01/01/1974

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
A VITERBI DECODER

by

L. van de Meeberg
A Viterbi decoder

by

L. van de Meeberg

TH-Report 74-E-49
October 1974
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>3</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>4</td>
</tr>
<tr>
<td>2 THE ENCODER</td>
<td>5</td>
</tr>
<tr>
<td>3 THE VITERBI ALGORITHM</td>
<td>6</td>
</tr>
<tr>
<td>3.1 The algorithm in terms of the trellis diagram</td>
<td>6</td>
</tr>
<tr>
<td>3.2 Analytical representation of the algorithm</td>
<td>7</td>
</tr>
<tr>
<td>3.3 Some comments on the metric and path registers</td>
<td>8</td>
</tr>
<tr>
<td>4 THE GENERATING FUNCTIONS</td>
<td>9</td>
</tr>
<tr>
<td>5 ERROR PROBABILITIES</td>
<td>11</td>
</tr>
<tr>
<td>5.1 The error event probability</td>
<td>11</td>
</tr>
<tr>
<td>5.2 The bit error probability</td>
<td>12</td>
</tr>
<tr>
<td>5.3 Generally valid upper bounds for $P_E$ and $P_B$</td>
<td>12</td>
</tr>
<tr>
<td>6 REALIZATION OF THE ENCODER</td>
<td>15</td>
</tr>
<tr>
<td>6.1 The metric registers</td>
<td>15</td>
</tr>
<tr>
<td>6.2 The path registers</td>
<td>16</td>
</tr>
<tr>
<td>6.3 The control unit</td>
<td>17</td>
</tr>
<tr>
<td>6.4 The error detector</td>
<td>17</td>
</tr>
<tr>
<td>7 MEASUREMENTS</td>
<td>21</td>
</tr>
<tr>
<td>8 SUGGESTIONS FOR SIMPLIFYING THE CIRCUIT</td>
<td>23</td>
</tr>
<tr>
<td>8.1 Simplification of the metric</td>
<td>23</td>
</tr>
<tr>
<td>8.2 Large-scale integration</td>
<td>23</td>
</tr>
<tr>
<td>Appendix 1</td>
<td>24</td>
</tr>
<tr>
<td>Appendix 2</td>
<td>25</td>
</tr>
<tr>
<td>Appendix 3</td>
<td>26</td>
</tr>
<tr>
<td>Bibliography</td>
<td>28</td>
</tr>
<tr>
<td>Acknowledgement</td>
<td>28</td>
</tr>
</tbody>
</table>
Summary

After an introduction to the structure of convolutional codes, this report discusses a Viterbi decoder for the simplest non-systematic convolutional code (constraint length $K = 3$).

In Section 1 it is shown how this code is generated. The consecutive data bits, which are to be encoded, are shifted into a 3-bit shift register. To this register two mod-2 adders are connected; one is linked to all three stages and the other to the first and the last stage of the shift register. The sums presented by the mod-2 adders appear alternately at the output terminal of the encoder. In this way a binary code with rate $1/2$ is obtained. It is shown that all possible code sequences can be represented as paths in a so-called trellis diagram.

Section 2 deals with the Viterbi algorithm, an optimal algorithm for maximum-likelihood decoding of convolutional codes. When a sequence of code digits is mutilated in a transmission channel, the particular path through the trellis diagram is searched which has the smallest Hamming distance with respect to the received sequence. In this case maximum-likelihood decoding is equivalent to minimum-distance decoding.

In Section 3 the generating function is derived. Analysis of the function yields all distance information of the code.

In Section 4 it is shown that the error correction obtained with this algorithm is quite considerable. The bit error probability can be calculated by computing the probability of a wrong path being followed and ascertaining the number of bit errors caused thereby. The generating function appears to be a most useful tool for formulating this error probability. It is derived that in the case of the encoder and decoder being linked by a binary symmetric channel, the bit error probability remains below approximately $50\, p^2$, where $p$ denotes the cross-over probability of this channel.

The circuit of a decoder making use of TTL is discussed in Section 5.

Measurements dealt with in Section 6 show that the actual bit error rate is in very close agreement with the calculations of Section 4.

In Section 7 some possibilities are suggested for simplifying the circuit by means of large-scale integration.
I INTRODUCTION

Since Claude Elwood Shannon published his "Mathematical Theory of Communications" in the Bell System Technical Journal in 1948, an immense amount of work has been done with the object of improving the reliability of communications. Information theory has had great influence on the development of digital modulation systems, space and satellite communications and such fields as semantics, psychology and genetics. However, the most important activities in the discipline of information theory concern source coding and channel coding, the latter of which is used for reducing the effects of noise introduced in a communication channel. In general, a channel encoder adds redundancy to an information source, and it is this redundancy which enables the decoder to improve the signal-to-noise ratio or to lower the error rate.

The class of error-correcting channel codes known as convolutional codes offers promising practical possibilities; the encoding and decoding techniques are considerably simpler than is the case with block codes.

Several methods have been devised for decoding convolutional codes, such as sequential decoding (Wozencraft [1], [2]), threshold decoding (Massey [3]), the Fano algorithm [4] and recently the Viterbi algorithm ([5] to [9]). The present report discusses the design of a decoder based on the last of these for the simplest convolutional code. By way of introduction, in this introductory section the general form of a convolutional encoder will first be discussed, and then some subclasses of convolutional codes will be dealt with in brief.

A convolutional encoder is a linear "finite-state machine" consisting of a $K$-stage shift register and $n$ mod-2 adders. The data sequence, which is usually binary, is shifted into the register $b$ bits at a time $(b < n)$. In general, such an encoder will thus assume the form depicted in Fig.1.1, which has been drawn from right to left to show the bits in their correct sequence (first bit at the left). The rate $R$ of such a code is $b/n$, all $b$ data bits being converted into $n$ code digits. This reduction in rate is the price that has to be paid for the error-correcting feature. We shall confine ourselves to codes of which $b = 1$, thus with rate $1/n$. The systematic convolutional codes form a subclass of these codes. In a systematic encoder $n - 1$ mod-2 adders are connected to the stages of the shift register, whilst the $n$th adder is replaced by the direct connection to the first stage.

Bucher and Heller [10] showed that for high values of $K$ the behaviour of a systematic encoder of constraint length $K$ is substantially the same as that of a non-systematic encoder of constraint length $K (1 - R)$. For this reason we shall confine ourselves to non-systematic codes. A problem peculiar to non-systematic codes is that of catastrophic error propagation: with certain connection patterns between the mod-2 adders and the shift register, it is possible for a finite number of errors in transmission to give rise to an infinite number of errors in the decoded data sequence. Massey and Sain [11] showed that a rate $1/n$ convolutional code is subject to catastrophic error propagation if and only if the subgenerator polynomials contain a common factor. Applying this criterion, Rosenberg [12] showed that only a small fraction of the non-systematic codes, viz. $1/(2^n - 1)$, is in fact catastrophic. Therefore, the question of catastrophic error propagation will not be further dwelt upon here.

In designing convolutional encoders the main problem consists in finding the optimal connections between the mod-2 adders and the shift register. The main criterion to be kept in mind is the minimum free distance, which should be as large as possible. In Section 5 the meaning of this will be explained. Optimal connections have been ascertained up to a constraint length $K = 9$, i.e. by Oldenwalder [13]. Here, we shall confine our attention to a code with $K = 3$, $n = 2$ and a minimum free distance 5. This code belongs to the class of complementary convolutional codes, which again form a subclass of the non-systematic codes. This implies that both mod-2 adders are connected with the first as well as with the last stage of the shift register.

![Fig.1.1 General representation of a convolutional encoder.](image-url)
The simplest non-systematic convolutional code is generated in the following way, as illustrated by Fig.2.1. Two mod-2 adders are connected to a three-stage shift register. The outputs $Q_1$ and $Q_2$ of these adders are alternately connected to the output terminal of the encoder (first $Q_1$, then $Q_2$).

Denoting the three positions of the shift register by $S_0$, $S_1$ and $S_2$ respectively, yields the truth table below (Table I).

The transitions of the states will be investigated on the basis of the table. The four states which the positions $S_1$ and $S_0$ can assume are denoted by $a$, $b$, $c$ and $d$. This provides the transitions given in Table 2.

This may be illustrated by the following example. Assume the data sequence to be 1 1 0 1 0 and the shift register to contain initially three zeros (state $a$). When the first bit, a 1, is shifted in, the contents of the shift register will become 0 0 1 (state $b$), so that $Q_1 = 0 \oplus 0 \oplus 1 = 1$ and $Q_2 = 0 \oplus 1 = 1$, hence 1 1 appears at the output. Now the next bit is shifted in, again a 1. The shift register then contains 0 1 1 (state $d$). Therefore, $Q_1$ becomes 0 and $Q_2$ becomes 1. At the next step the register contains 1 1 0, whence $Q_1 = 0$ and $Q_2 = 1$. Proceeding in this way we find for the data sequence 1 1 0 1 0 the code 1 1 0 1 0 1 0 1 0.

Each data bit is converted into two code digits which are fed into the channel, so the rate is $\frac{1}{2}$.

The linearity of this system can easily be demonstrated by comparing the response to two different data sequences with the response to their mod-2 sum:

$$0 0 0 0 0 0 \ldots \rightarrow 1 1 0 1 1 0 0 0 0 0 0 0 \ldots$$
$$0 0 1 0 0 0 \ldots \rightarrow 0 0 0 0 1 1 0 1 1 0 0 0 0 0 0 0 \ldots$$
$$1 0 1 0 0 0 \ldots \rightarrow 1 1 0 0 0 1 0 1 1 0 0 0 \ldots$$

We shall now investigate the transitions of the states (see Table 2) more closely. To this end the states $a$, $b$, $c$ and $d$ are represented as 4 levels in a trellis diagram; an entered bit is represented by a solid line if it is a 0 or by a dashed line if it is a 1. The relevant two code bits are indicated along these lines, which thus represent the transitions. Moreover, we shall assume the shift register to contain initially three zeros, i.e. to be in state $a$. The trellis diagram will then be as shown in Fig.2.2.

### Table 1

<table>
<thead>
<tr>
<th>$S_2$</th>
<th>$S_1$</th>
<th>$S_0$</th>
<th>state</th>
<th>$Q_1$</th>
<th>$Q_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$a$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>$b$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$c$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$d$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>$a$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$b$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>$c$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$d$</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 2

If $S_1$ and $S_0$ are in state: $a$, $b$, $c$, $d$ and the shifted-in bit is: 0, 1, 0, 1, 0, 1, 0, 1 then $S_1$ and $S_0$ assume state: $a$, $b$, $c$, $d$ whilst $Q_1$ and $Q_2$ become: 0, 1, 1, 0, 0, 1, 1, 0.
3 THE VITERBI ALGORITHM

3.1 The algorithm in terms of the trellis diagram

The Viterbi algorithm is based on the principle of maximum-likelihood decoding, which in the present case is equivalent to minimum distance decoding. Upon reception of a sequence of bits, the particular path through this diagram will be searched which is closest to this sequence in the sense of Hamming distance; i.e., the path which differs from the received sequence by the minimum number of symbols. An example will make the meaning of this clear.

It was explained in the previous section that the data sequence 1 1 0 1 0 is converted into the code 1 1 0 1 0 0 1 0. Assume this code sequence to be mutilated in the transmission channel so that 1 0 0 1 1 1 0 0 1 0 is received, in other words that the second and fifth bits are erroneous. The paramount question is how this received sequence will be decoded.

Let us first consider the first pair of bits, 1 0. Starting at the left top of the trellis diagram, we see that only two paths start from this point, viz. the path \(a - a (0 0)\) and the path \(a - b (1 1)\). Both paths are at a Hamming distance 1 from the first pair of received bits (1 0). We keep these distances and both paths in mind. Path \(a - a\) corresponds to a 0 and path \(a - b\) to a 1 in the relevant data sequence.

Now we consider the second pair of received bits, 0 1. From the reached point \(a\) there is one path \((0 0)\), again to \(a\), thus at a Hamming distance 1 from 0 1. Since the first step already involved a Hamming distance 1, the path \(a - a\) \((0 0 0 0)\) is at a total Hamming distance 2 from the received bits (1 0 1 1 0 1).

From point \(a\) reached after the first step, there is also a path \((1 1)\) to \(b\), likewise at a distance 1 from the second pair of received bits, thus this path \(a - a - b (0 0 1 1)\) is also at a total distance 2 from the first two received bits.

From \(b\) there is a path \((1 0)\) to \(c\) at a distance 2 to 0 1, totalling 1 + 2 = 3. In other words, the path \(a - b - c (1 1 1 0)\) is at a distance 3 from 1 0 0 1.

Finally, there is a path \(b - d (0 1)\) at distance 0 from 0 1, yielding a total distance of 1 + 0 = 1. The path \(a - b - d (1 1 0 1)\) is thus at a distance 1 from 1 0 0 1.

Summarizing, we have

- a path terminating in \(a\) at a total distance 2 (metric 2)
- a path terminating in \(b\) with metric 2
- a path terminating in \(c\) with metric 3
- a path terminating in \(d\) with metric 1.

These four paths correspond to the data sequences 0 0, 0 1, 1 0 and 1 1 respectively.

The situation becomes slightly more complex at the next step because each of the nodes \(a, b, c\) and \(d\) is now the terminus of two paths. We need only store the one with the smallest metric. The other path can be disregarded because it has a larger metric and is thus less probable. If the metric values of both paths are identical, we make an arbitrary choice by flipping a coin; in this example, let us suppose that this means disregarding the lower of the two paths.

The third pair of bits received is 1 1. We now consider Fig. 3.1 in which the metric values after the second step are indicated at the left.

The Hamming distance of path \(a - a\) is 2 (metric 2 + 2 = 4), that of path \(c - a\) is 0 (metric 3 + 0 = 3). Path \(a - a\) can thus be disregarded and path \(c - a\) with metric 3 should be stored.

The Hamming distance of path \(a - b\) is 0 (metric 2 + 0 = 2), that of path \(c - b\) is 2 (metric 3 + 2 = 5). Only path \(a - b\) with metric 2 need be stored.

The Hamming distance of path \(b - c\) is 1 (metric 2 + 1 = 3), and so is that of path \(d - c\) (metric 1 + 1 = 2). Only path \(d - c\) with metric 2 is stored.

Finally, the Hamming distance of path \(b - d\) is 1 (metric 2 + 1 = 3), and so is that of path \(d - d\) (metric 1 + 1 = 2). Only path \(d - d\) with metric 2 is stored.

Recapitulating, the new metric values at points \(a, b, c\) and \(d\) are now 3, 2, 2 and 2, respectively, and the paths stored correspond to the data sequences 1 0 0 0 0 1, 1 1 0 and 1 1 1, respectively.

For the next two steps we proceed in the same way. After the fourth step we find the metric values 3, 2, 3, 3 and the paths corresponding to the data sequences 1 0 0 0 0, 1 1 0 1, 0 0 1 0 and 0 0 1 1, respectively. The fifth and last step yields the metric values 4, 4, 2, 3 and the paths corresponding to the data sequences 1 0 0 0 0, 1 0 0 0, 1 1 0 1 0 and 0 0 1 1, respectively.

Fig. 3.2 gives the resulting trellis diagram, omitting the disregarded paths and showing the metric values after each step. It is seen that after reception of the sequence 1 0 0 1 1 1 0 0 1 0 the path terminating in \(a\) (data sequence 1 0 0 0 0) has a metric 4, and so has the path terminating in \(b\) (data sequence 1 0 0 0 1). The path terminating in \(c\) (data sequence 1 1 0 1) has a metric 2 and that terminating in \(d\) (data sequence 0 0 1 1) has a metric 3.

The path with the smallest metric value (2) with respect to the received bit sequence appears to be \(a - b - d - c - b - c\), corresponding to the data sequence 1 1 0 1 0 and the code sequence 1 1 0 1 0 1 0 0 1 0. It is seen to be identical to the code sequence generated by the encoder, two bits of which were mutilated in the transmission channel. The errors have
thus been corrected. It should, however, be realized that all
errors will not necessarily be corrected by this decoding
algorithm. If the error rate is higher, or if the error distribu-
tion is different, there is a chance of the wrong path being
chosen. This will be explained by the following example.

Assume that the last bit of the mentioned sequence is also
wrongly received, so that the received sequence becomes
1 0 0 1 1 1 0 0 1 0, thus containing a third error. The paths
through the trellis diagram then become as shown in Fig.3.3.
After the last step the metrics will then be 3, 3, 3, and
the corresponding data sequences 0 0 1 1 0 0 1 0, 1 1 0 1 0
and 1 1 0 1 1 respectively. Since the metric
values of all paths are 3, the choice is again arbitrary, the
four paths being equally likely. If the first is chosen, there
will be 4 bit errors; the second path results in 3 bit errors,
the third path yields the correct data sequence, and if the
fourth is chosen there will be 1 bit error.

For the metric values we shall use the symbols $M_a^n$, $M_b^n$, $M_c^n$ and $M_d^n$, where the $n$ stands for the order of the step
(or the instant). The symbol $x_n$ denotes the $n$th pair of
bits, and the symbols $x_{n1}$ and $x_{n2}$ the corresponding
individual bits. For the decoded sequence we shall use the
symbol $P$, defined by the same indices as $M$. Finally,
the Hamming distance between $x_n$ and, for example 0 0, will
be written $D (x_n - 0 0)$.

The first step is thus expressed by:

$M_a^1 = D (x_1 - 0 0)$, $P_a^1 = 0$
$M_b^1 = D (x_1 - 1 1)$, $P_b^1 = 1$

and the second step by:

$M_a^2 = M_a^1 + D (x_2 - 0 0)$, $P_a^2 = P_a^1$, $0 = 0 0$
$M_b^2 = M_b^1 + D (x_2 - 1 1)$, $P_b^2 = P_b^1$, $1 = 0 1$
$M_c^2 = M_b^1 + D (x_2 - 1 0)$, $P_c^2 = P_b^1$, $0 = 1 0$
$M_d^2 = M_b^1 + D (x_2 - 0 1)$, $P_d^2 = P_b^1$, $1 = 1 1$

whilst the third step becomes:

$M_a^3 = \min \{ M_a^2 + D (x_3 - 0 0), M_c^2 + D (x_3 - 1 1) \}$
$M_b^3 = \min \{ M_b^2 + D (x_3 - 1 1), M_c^2 + D (x_3 - 0 0) \}$
$M_c^3 = \min \{ M_b^2 + D (x_3 - 1 0), M_d^2 + D (x_3 - 0 1) \}$
$M_d^3 = \min \{ M_b^2 + D (x_3 - 0 1), M_d^2 + D (x_3 - 1 0) \}$

if $M_a^3 = M_a^2 + D (x_3 - 0 0)$, then $P_a^3 = P_a^2$, $0 = 0 0$
if $M_a^3 = M_c^2 + D (x_3 - 1 1)$, then $P_a^3 = P_c^3$, $0 = 1 0$
if $M_b^3 = M_b^2 + D (x_3 - 1 1)$, then $P_b^3 = P_b^2$, $1 = 0 1$
if $M_b^3 = M_c^2 + D (x_3 - 1 0)$, then $P_b^3 = P_c^3$, $0 = 1 0$
if $M_c^3 = M_b^2 + D (x_3 - 0 1)$, then $P_c^3 = P_d^3$, $0 = 1 1$
if $M_d^3 = M_b^2 + D (x_3 - 1 0)$, then $P_d^3 = P_d^2$, $1 = 1 1$

The procedure for the following steps is identical to that
for the third step, so that, in general, for $n \geq 3$ the follow-
ing scheme applies:

The validity of these relationships is general (they also
apply to $n = 1$ and $n = 2$) if we put $M_a^0 = 0$, $M_b^0 = R$,
$M_c^0 = S$ and $M_d^0 = T$, where $R > 3$, $S > 2$ and $T > 3$.

This can be demonstrated by carrying out the first three
steps in the algorithm with these initial values for all pos-
sible received bit sequences. As a result of the minimizing
procedure, $R$, $S$ and $T$ will disappear from the $M^3$ values
after the third step.
3.3 Some comments on the metric and path registers

Up to now it has been assumed that the initial state was $a$; in other words, that the shift register initially contained three zeros. In general, this will not be the case. If we drop this assumption and start at an arbitrary node in the trellis diagram, then all four initial metric values are taken to be zero. It is further obvious that if an arbitrary codeword has been correctly received up to a certain instant $i$, only four combinations of metric values are possible, viz.

$M_d^i = 0$  $M_d^i = 2$  $M_d^i = 3$  $M_d^i = 3$

$M_d^i = 2$ or $M_d^i = 0$ or $M_d^i = 3$ or $M_d^i = 2$

$M_d^i = 3$  $M_d^i = 3$  $M_d^i = 3$  $M_d^i = 0$

It should also be recognized that for each step only the mutual differences between the metrics are of importance. In designing a decoder, advantage may be taken of this feature by always deducting from all metric values the smallest one, $M_m$. In this way we prevent the metric values from becoming excessive (the bit sequences may be very long). By this procedure the total number of possible metric combinations is limited to 31, as can be ascertained by taking 4 arbitrary metric values as a starting point and comparing all possible sequences and the corresponding metrics. These are entered in Table 4.

This table shows that the highest occurring metric value is 3. Four two-bit memories are therefore sufficient for storing the metric values. The metric calculations may be performed by a simple combinatorial network or even a PROM or a PLA.

As far as the path registers are concerned, it should be noted that $P_d^i$ always terminates with 0 0, $P_d^i$ with 0 1, $P_d^i$ with 1 0 and $P_d^i$ with 1 1, as clearly shown by the trellis diagram. For example, to get from an arbitrary node to $d$ in two steps, it is always necessary to travel along two dashed lines, which means that $P_d^i$ always ends in two ones.

It is also of interest to know how long the shift registers must be to store the four paths. If we consider that messages may consist of many thousands of bits, it will become obvious that it would be unpractical to wait until an entire message has been received before starting to read out the path with the smallest metric value. Fortunately, after a number of steps, the first bits of the four stored paths will coincide. These bits can then be read out, as will become clear from the following example.

Reverting to the trellis diagram shown in Fig.3.2, let us assume that the next bits received are 0 1 0 1. Omitting the paths which come to a dead end, extension of the diagram by the following two steps then gives the diagram shown in Fig.3.4. The diagram can now be simplified to that of Fig.3.5 by again omitting the paths which can be deleted or come to a dead end. This diagram shows that all four paths have the first bit (1) in common. Therefore, this bit can safely be read out.

However, there is no certainty that after the next step all paths will again have a bit in common. Therefore, if for each path we were to store only six bits in a six-bit shift register and, after each step, read out that overflow bit which had the smallest metric value of the four, we would risk introducing additional bit errors due to premature truncation of the stored paths.

It is difficult to evaluate this path memory truncation error. Jacobs and Heller [6] ascertained that a path register with a length of 4 to 5 times the constraint length of the coder would as a rule be sufficient to avoid such errors. In the present case this would amount to a path register length of 12 to 15 bits. The influence of this length on the bit error probability will be discussed in Section 7 for several values of the error probability in a binary symmetric channel.

<table>
<thead>
<tr>
<th>$M_d^i$</th>
<th>0 1 0 0</th>
<th>0 0 1</th>
<th>0</th>
<th>0 1 2 1</th>
<th>0 1 2 0 2 2 2 0 2 3 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_d^i$</td>
<td>0 0 1</td>
<td>0 1 0</td>
<td>0 1 0</td>
<td>0 1 1 1 0 2 2 1 2 2 0 2 2 2 0 2 3 3</td>
<td></td>
</tr>
<tr>
<td>$M_d^i$</td>
<td>0 0 0 1</td>
<td>0 1 1 1 2 0 2</td>
<td>1 1 0 2 2 2 0 2 2 0 2 3 3 0 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_d^i$</td>
<td>0 0 0 0 1 0 1 1 0 1 2 1 0 0 1 1 2 0 2 2 1 2 2 2 0 2 3 3 2 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Fig. 3.4 Trellis diagram after reception of 4 more bits.

Fig. 3.5 Four paths with common first bit.
4 THE GENERATING FUNCTIONS

To calculate error probabilities it is first necessary to investigate the distance properties of the code. Convolutional codes are group codes, which implies that the set of distances of the all-zeros codeword to all other codewords is equal to the set of distances of any specific codeword to all others. It will therefore be useful to ascertain how many paths deviate from the all-zeros path, at what distance each of these paths is located, and how many bit errors each path represents.

All this information can be expressed in terms of the so-called generating function, which we shall now derive. As a starting point we shall reconsider the state diagram shown in Fig.2.3. Since we are interested in the paths which deviate from the all-zeros path and merge with it again later, we cut this diagram open at node \(a\), so that it assumes the form shown in Fig.4.1. The distance of each branch to the all-zeros codeword will be denoted by an exponent of a formal variable \(D\), so that the branch \(a-b\) for example will be labeled \(D^2\), the Hamming distance between 11 and 00 being 2. Fig.4.1 will thus become as shown in Fig.4.2.

We shall now investigate how many ways there are to pass through the diagram. Let us first consider the upper part, redrawn in Fig.4.3. This can be passed through in

\[
D + D^2 + D^3 + \ldots = D/(1-D) \text{ ways. (Since } D \text{ is defined in the neighbourhood of 0, this summation is permissible.)}
\]

The diagram of Fig.4.2 may therefore be simplified to that of Fig.4.4.

We thus have the following possibilities of travelling through the whole diagram:

\[
\frac{D^5}{1-D} + \frac{D^6}{(1-D)^2} + \frac{D^7}{(1-D)^3} + \ldots = \frac{D^5/(1-D)}{1-D/(1-D)} = \frac{D^5}{1-2D}
\]

This expression is called the generating function \(T(D)\), being:

\[
T(D) = \frac{D^5}{1-2D} = D^5 + 2D^6 + 4D^7 + \ldots + 2^kD^{k+5}, \quad (4.1)
\]

in which \(k = 0, 1, 2, \ldots\). This expression thus simply indicates that there is one path at distance 5 from the all-zeros path, two at distance 6, and so forth. In general there are \(2^k\) paths at distance \(k+5\).

We shall now also express in the generating function the lengths of the paths and the number of ones in each path (hence the number of bit errors if the zero code word is transmitted). To this end we label each branch in the diagram of Fig.4.1 with an \(L\) and add an additional label \(N\) to the branches which indicate a data-one (i.e. the branches in dashed line). The diagram of Fig.4.5 thus obtained then contains all information.
Analysing this diagram in the same way as Fig. 4.2 yields:

\[
T(D, L, N) \triangleq \frac{D^5 L^3 N}{1 - DL (1 + L)N} =
\]

\[
= D^5 L^3 N + D^5 L^4 (1 + L)N^2 +
\]

\[
\ldots + D^5 L(1 + L)k N^k + k + \ldots
\]

where \( k \) is again 0, 1, 2, 3, \ldots. This expression has the following meaning.

There is one path at distance 5 of length 3 in which 1 data-one occurs; there are two paths at distance 6, viz. one of length 4 and one of length 5, two data-ones occurring in both paths, and so forth.

If, say, only \( D \) and \( N \) are of interest, \( L \) is put equal to unity in eq. (4.2), which gives:

\[
T(D, N) \triangleq \frac{D^5 N}{1 - 2DN} =
\]

\[
= D^5 N + 2D^6 N^2 +
\]

\[
\ldots + 2^kD^k + 5N^k + 1 + \ldots,
\]

\( (k = 0, 1, 2, \ldots) \) (4.3)

These generating functions will be required in the next section for determining the several error probabilities.

The generating function \( T(D) \) can also be derived in a more general way by means of the distance matrix, which indicates the weight necessary to change over from one state to another (or to the same state) in \( n \) steps. The one-step matrix for the middle part of Fig. 3.2 thus becomes:

\[
\Delta = I + \begin{pmatrix} 0 & D & D \\ 1 & 0 & 0 \end{pmatrix}
\]

and the two-step matrix: \( \Delta^2 = \begin{pmatrix} D & D^2 & D^2 \\ 0 & D & D \\ 0 & D & D \end{pmatrix} \)

whilst \( \Delta^0 + \Delta^1 + \Delta^2 + \ldots = I + \Delta + \Delta^2 + \ldots = (I - \Delta)^{-1} \). We are interested only in \( b \) \( c \), i.e. in the first row and the second column of the matrix \( (I - \Delta)^{-1} \):

\[
(I - \Delta)^{-1} = \begin{pmatrix} \frac{D}{1 - 2D} \\ \frac{D}{1 - 2D} \end{pmatrix}
\]

For the whole diagram of Fig. 4.2 we then get

\[
T(D) = \frac{D^5}{1 - 2D}
\]

Both \( T(D, N) \) and \( T(D, L, N) \) can be calculated in an analogous way. For more complex convolutional codes this method is in fact preferable to the previous one which is apt to become very time-consuming.
5 ERROR PROBABILITIES

5.1 The error event probability

The error event probability is understood to be the probability of, at a certain node in the trellis diagram, an erroneous path being chosen which merges again with the correct path for the first time at that node.

It follows clearly from the trellis diagram that the shortest path which deviates from the all-zeros path is the path a-b-c-a (1 1 1 0 1 1), corresponding to the data sequence 1 0 0. It is situated at distance 5 (termed the minimum free distance) from the all-zeros path.

Let us introduce the symbol $p$ to denote the probability of a 1 being received when a 0 has been transmitted via a binary symmetric channel (BSC) or vice versa. The probability of a bit being correctly received thus amounts to $1 - p = q$ (see Fig. 5.1). We shall now ascertain the probability of this path at distance 5 being chosen if the all-zeros codeword has been transmitted.

![Fig. 5.1 Representation of a binary symmetric channel.](image)

There are $\binom{5}{3}$ possible combinations of 2 zeros and 3 ones in the positions 1, 2, 3, 5 and 6 of a sequence of 6 bits. The 4th bit can be disregarded because it is a 0 in both sequences and therefore does not contribute to the probability of an erroneous path being chosen. These $\binom{5}{3}$ sequences are all at distance 2 from the path 1 1 1 0 1 1 and at distance 3 from the all-zeros path. The probability of the path 1 1 1 0 1 1 being chosen instead of the correct 0 0 0 0 0 0 if 3 of the 5 bits are not correctly received thus amounts to $\binom{5}{3} p^3 q^3$.

An analogous argument applies to the $\binom{5}{4}$ possible combinations of 1 zero and 4 ones and to the sequence of 5 ones. The total probability of the erroneous path being chosen is thus

$$P_5 = \sum_{e=3}^{5} \binom{5}{e} p^e q^5 - e.$$

A similar equation can be derived for any path at an odd distance, so that, in general,

$$P_k = \sum_{e=(k+1)/2}^{k} \binom{k}{e} p^e q^{k-e}, \quad (k \text{ is odd}).$$

There is also a path in the trellis diagram at distance 6 from the all-zeros path, e.g. a-b-d-c-a (1 1 0 1 0 1 1); this corresponds to the data sequence 1 1 0 0, hence of length 4.

This path will be chosen if 4 or more of the bits in the positions 1, 2, 4, 6, 7 and 8 of the sequence 0 0 0 0 0 0 0 0 are not correctly received. If exactly 3 of these 6 bits are erroneous, the correct and the erroneous path will both be at distance 3 from the received sequence. The probability of the erroneous path then equals the probability of the correct path being chosen and thus amounts to $1/2$. The total probability of a path at distance 6 being chosen is therefore

$$P_6 = \frac{1}{2} \left( \binom{6}{3} p^3 q^3 + \sum_{e=4}^{6} \binom{e}{k} p^e q^{6-e} \right).$$

In general, for a path at an even distance,

$$P_k = \frac{1}{2} \left( \binom{k}{2} p^{k/2} q^{k/2} \right) + \sum_{e=k/2+1}^{k} \binom{k}{e} p^e q^{k-e} \quad (k \text{ is even})$$

According to the generating function $T(D)$, there is one path at distance 5, two paths at distance 6, and in general $2^k$ paths at distance $k + 5$ (cf. Eq.(4.1)). It is hardly feasible to calculate the probability of one of the many erroneous paths being chosen at any given node; we can say, however, that this probability is in any case smaller than the sum of the probabilities for any possible path, as given by

$$P_E < P_5 + 2P_6 + 4P_7 + \ldots + 2^k P_{k+5} + \ldots, \quad (k = 0, 1, 2, \ldots)$$

In Appendix 1 it is demonstrated that $P_5 = P_6$, $P_7 = P_8$ ... and, in general, that $P_k = P_{k-1}$ for even values of $k$. In Appendix 2 it is derived that

$$P_k < \frac{5}{32} (2\sqrt{p})^k,$$

whence we may write:

$$P_E < 3P_5 + 12P_8 + \ldots + 3x4kP_{2k+6} + \ldots, (k = 0, 1, 2, \ldots)$$

$$\frac{5}{32} \times 3 \left( (2\sqrt{p})^6 + 4(2\sqrt{p})^8 + 4k(2\sqrt{p})^{2k+6} + \ldots \right)$$

$$= \frac{15}{32} \left( 1 - 4(2\sqrt{p})^2 \right) = \frac{15}{32} \frac{64p^3}{1 - 16p} = \frac{30p^3}{1 - 16p}, \quad (5.3)$$

provided that $p < 1/16$. 
5.2 The bit error probability

The bit error probability $P_E$ is defined as the ratio of the expected number of bit errors in the decoded data sequence to the total number of bits transmitted. From the generating function

$$T(D, N) = D^5 N + 2 D^6 N^2 + \ldots + 2^k D^{k+5} N^{k+1} + \ldots,$$

$$k = 0, 1, 2, \ldots$$ (4.3)

it follows that there are $2^k$ paths at distance $k + 5$, each of which corresponds to $(k + 1)$ ones in the original data sequence. The exponents of $N$ thus determine the number of bit errors per path. To obtain these exponents as weighting factors before each term, the function $T(D, N)$ should be differentiated with respect to $N$. Subsequently $N$ can be eliminated again from the derivative by putting $N = 1$; thus,

$$\frac{dT(D, N)}{dN} \bigg|_{N = 1} = D^5 + 2 x 2 D^6 + \ldots + \binom{k+1}{k} 2^k D^{k+5} + \ldots,$$

$$k = 0, 1, 2, \ldots$$

In a similar way as for $P_E$ we find for the bit error probability:

$$P_B < P_E < 2 x 2 P_E + \ldots + \binom{k+1}{k} 2^k P_E + \ldots,$$

$$(k = 0, 1, 2, \ldots)$$

and with $P_k = P_{k-1}$ for even values of $k$ and $P_k < \frac{5}{32} (2\sqrt{p})^k$:

$$P_B < 5 P_E + 4 x 11 P_E + 4k(6k+5) P_{2k+6} + \ldots,$$

$$(k = 0, 1, 2, \ldots)$$

provided that $p < 1/16$.

5.3 Generally valid upper bounds for $P_E$ and $P_B$

Viterbi [5] calculated different upper bounds by demonstrating that

$$P_k < \left(2 \sqrt{p(1-p)} \right)^k,$$

from which it can easily be derived that

$$P_E < \frac{T(D)}{D = 2 \sqrt{p(1-p)}} = \left(2 \sqrt{p(1-p)} \right) \frac{5}{4} \left(1 - 4 \sqrt{p(1-p)} \right)^k,$$

(5.5)

By making use of the relations $P_k = P_{k-1}$ for even $k$ and $P_k < \Gamma(2 \sqrt{p})^k$ where $\Gamma$ is a constant which is determined by the minimum free distance of the used convolutional code (in the case under consideration $\Gamma = 5/32$), it is possible to derive tighter bounds, as shown in Appendix 3 and previously by van de Meeberg [14]:

$$P_B < \frac{dT(D, N)}{dN} \bigg|_{N = 1} = \frac{2 \sqrt{p}(1-p)}{1 - 4 \sqrt{p}(1-p)}.$$

(5.6)
It is true that these expressions are less compact than those derived by Viterbi, but the upper bounds given by eqs (5.7) and (5.8) are considerably tighter. For the sake of comparison the upper bounds given by the eqs (5.6) and (5.8) have been plotted in Fig.5.2 as functions of the channel crossover error probability \( p \). For small values of \( p \) the bound according to eq.(5.8) asymptotically approaches 50 \( p^3 \), whereas the bound according to eq.(5.6), as derived by Viterbi, approaches 32 \( p^{2\frac{1}{3}} \). Hence the smaller the value of \( p \), the greater will be the difference between the two asymptotes.

In the graph of Fig.5.2 the measured error rate curve has also been plotted; how the measurements were made is discussed in Section 6.
Fig. 6.1 Circuit of the metric registers.
6 REALIZATION OF THE DECODER

The design of the decoder was based on the use of TTL MSI-circuits (Fairchild 9300 and 9000 series), complemented where necessary by Texas Instruments circuits (7483 and 74164).

6.1 The metric registers

Fig. 6.1 (opposite) shows the circuit diagram of the metric registers. We shall discuss its operation with reference to Table 3.

Assume the values $M_{n-1}$ to be present at the $Q$ outputs of the four shift registers 9300 ($A$, $B$, $C$ and $D$). (The $Q_1$ outputs are followed by exclusive ORs connected as buffers to cope with the low input resistance of the $A_3$ inputs of the full-adders.) In the eight full-adders 7483 these values are now added to the Hamming distances of the $n$th pair of received bits $x_n$ to 0 0, 1 1, 1 0 and 0 1, grouped according to the formulae.

By way of example, the sum of $M_{n-1}$ and the Hamming distance from $x_n$ to 0 0 appears at the output of the full-adder A00. The Hamming distances are determined by the circuit of Fig.6.2; the truth table and switching functions are given in Tables 5 and 6 respectively.

The outputs of the full-adders are subsequently compared two by two and multiplexed. The $A < B$ outputs of the comparators 9324 ($A$, $B$, $C$ and $D$) are linked to the select inputs of the multiplexers 9322 ($A$, $B$, $C$ and $D$) so that at their outputs the minima of the two presented full-adder outputs appear. Expressed in terms of the formulae in Table 3, this amounts to $M_d^n$, $M_b^n$, $M_e^n$ and $M_d^n$ appearing at the outputs of the multiplexers.

Subsequently the minimum of $M_d^n$ and $M_b^n$ and that of $M_e^n$ and $M_d^n$, and finally the minimum of these two minima are determined in the same way. We thus see that $M_{mn}^n$ is the minimum of $(M_d^n, M_b^n, M_e^n, M_d^n)$. As pointed out in Section 3.3, this value of $M_{mn}^n$ should be deducted from all four values of $M^n$.

Table 5

<table>
<thead>
<tr>
<th>$x_n1$</th>
<th>$x_n2$</th>
<th>$d-1$</th>
<th>$d$</th>
<th>$d-1$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6

<table>
<thead>
<tr>
<th>output</th>
<th>$i$</th>
<th>$j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d-1$</td>
<td>$x_{n1}$</td>
<td>$x_{n2}$</td>
</tr>
<tr>
<td>$d-0$</td>
<td>$x_{n1}$</td>
<td>$x_{n2}$</td>
</tr>
<tr>
<td>$d-1$</td>
<td>$x_{n1}$</td>
<td>$x_{n2}$</td>
</tr>
<tr>
<td>$d-1$</td>
<td>$x_{n1}$</td>
<td>$x_{n2}$</td>
</tr>
</tbody>
</table>

Fig. 6.2 Logic circuit for determining the Hamming distances.
This is achieved by the circuit connected to the output of multiplexer 9322 (M); it determines the twos complement of \( M_{m}^{n} \). The truth table and switching functions are given in Table 7.

<table>
<thead>
<tr>
<th>Table 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>9322 (M)</td>
</tr>
<tr>
<td>( Z_c )</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

The additional exclusive-OR has been provided to act as a buffer to cope with the four \( B_3 \)-inputs of the full-adders.

The twos complement of \( M_{m}^{n} \) is added to \( M_{d}^{n}, M_{b}^{n}, M_{c}^{n} \) and \( M_{p}^{n} \) by means of the full-adders 7483 \((A, B, C\) and \( D)\). The result is fed back to the \( P \)-inputs of the corresponding shift registers 9300 \((A, B, C\) and \( D)\), so that at the next clock pulse the new values \( M_{d}^{n} - M_{m}^{n} \) appear at the \( Q \) outputs. The next two bits \( x_{n+1} \) can now be shifted in and processed.

The initial values \( M_{0}^{n} \) can be set to zero at the start of the decoding process by the master reset inputs of the shift registers. This implies that we start at an arbitrary level of the trellis diagram, thus taking the initial state of the encoder to be unknown.

The \( A < B \) outputs \( a, b, c, d \) of the relevant four comparators can be used for loading the path registers, whilst the \( A < B \) outputs \( p, q, r \) of the three other comparators can serve for reading out these registers, as discussed in the next subsection.

6.2 The path registers

Let us first consider the outputs \( a, b, c \) and \( d \) of the metric circuit, to which the following relationships apply:

- if \( M_{d}^{n+1} + D(x_{n} - 0) < M_{c}^{n+1} + D(x_{n} - 1) \), then \( a = H \)
- if \( M_{d}^{n+1} + D(x_{n} - 0) \geq M_{c}^{n+1} + D(x_{n} - 1) \), then \( a = L \)
- if \( M_{d}^{n+1} + D(x_{n} - 1) < M_{c}^{n+1} + D(x_{n} - 0) \), then \( b = H \)
- if \( M_{d}^{n+1} + D(x_{n} - 1) \geq M_{c}^{n+1} + D(x_{n} - 0) \), then \( b = L \)
- if \( M_{b}^{n+1} + D(x_{n} - 1) < M_{d}^{n+1} + D(x_{n} - 0) \), then \( c = H \)
- if \( M_{b}^{n+1} + D(x_{n} - 1) \geq M_{d}^{n+1} + D(x_{n} - 0) \), then \( c = L \)
- if \( M_{b}^{n+1} + D(x_{n} - 0) < M_{d}^{n+1} + D(x_{n} - 1) \), then \( d = H \)
- if \( M_{b}^{n+1} + D(x_{n} - 0) \geq M_{d}^{n+1} + D(x_{n} - 1) \), then \( d = L \)

This may be expressed by Table 8, which shows how the path registers must be filled.

<table>
<thead>
<tr>
<th>Table 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>if ( a = H ) then ( P_{a}^{n} = P_{a}^{n-1}, 0 ) ( a = L ) then ( P_{a}^{n} = P_{a}^{n-1}, 0 )</td>
</tr>
<tr>
<td>if ( b = H ) then ( P_{b}^{n} = P_{b}^{n-1}, 1 ) ( b = L ) then ( P_{b}^{n} = P_{b}^{n-1}, 1 )</td>
</tr>
<tr>
<td>if ( c = H ) then ( P_{c}^{n} = P_{c}^{n-1}, 0 ) ( c = L ) then ( P_{c}^{n} = P_{c}^{n-1}, 0 )</td>
</tr>
<tr>
<td>if ( d = H ) then ( P_{d}^{n} = P_{d}^{n-1}, 1 ) ( d = L ) then ( P_{d}^{n} = P_{d}^{n-1}, 1 )</td>
</tr>
</tbody>
</table>

Table 9 holds for the outputs \( p, q, r \).

<table>
<thead>
<tr>
<th>Table 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

\( S_{0} \) and \( S_{1} \) are intended to serve as the select inputs of a 4-input multiplexer 9309, which reads out the path registers. The switching functions performed by the circuit shown in Fig.6.3 are

\( S_{0} = pq + qr \) and \( S_{1} = q \).

The algorithm offers some arbitrary choices which we shall deal with before describing the path register. It will therefore be useful to discuss Tables 8 and 9 in some greater detail.

Fig.6.3 Logic circuit for selecting the path registers.
the metric circuit only the $A < B$ outputs of the comparators 9324 ($A, B, C$ and $D$) are used (cf. Table 8).

This has been done for the sake of simplicity. Besides, for carrying out measurements with random signals it is immaterial which choice is made. However, this is not the case when the all-zeros codeword is transmitted, so in that case the upper of the two paths would be the best choice, whereas the lower path would be the best when the all-ones codeword is transmitted, because we then travel along the lower line of the trellis diagram. When the all-zeros codeword is transmitted the measured bit error rate is therefore likely to exceed the bit error rate measured with a random input signal; with the all-ones codeword the opposite is likely to be the case.

A similar arbitrary choice is possible when reading out the path registers. If 2, 3 or all 4 metric values are identical, a choice must be made between them. For the same reasons the circuit will then, too, choose the lower path (see Table 9).

For the sake of completeness some of the error probability measurements have been carried out not only with a random signal, but also with the all-zeros and the all-ones codewords.

Fig.6.4 shows the circuit diagram of the first of the four identical sections of the 16-bit path registers.

The 16 $Q$ outputs $P_a$ of the registers 9300 denoted by $A_1$, $A_2$, $A_3$, $A_4$ and the 16 $Q$ outputs $P_c$ of the registers 9300 denoted by $C_1$, $C_2$, $C_3$ and $C_4$ are linked to the inputs of the multiplexers 9322 (AI, A2, A3, A4). The 16 outputs are connected to the 16 $P$ inputs of the registers 9300 denoted by $A_1$, $A_2$, $A_3$, $A_4$. Depending on the select input, at the next clock pulse either $P_a$ or $P_c$ is read into $P_a$ in parallel (provided that $P_a = L$). If $a = H$, then $P_a^{t+1} = P_a^{t-1}$, whereas in the event of $a = L$, then $P_a^{t+1} = P_a^{t-1}$.

In order to comply with the first line of Table 8, a 0 must still be shifted into the register. To this end PE is required to be H, so that at the next clock pulse a 0 is indeed shifted in; it should be recognized that the JK input of register 9300 (A1) is grounded.

A similar argument applies to $P_b$, $P_c$ and $P_d$: however, a 1 is shifted into $P_b$ and $P_d$ at the second clock pulse.

The four NAND-gates 9009 (AD and BC) preceding the PE- and CP-inputs of the shift registers again serve as buffers.

The 4-input multiplexer 9309 connected to the last stage of each path register selects the sixteenth bit of the path register corresponding to the lowest metric value. The read-out information $S_0$ and $S_1$ is delayed by the latch 9314 (S); the delay must be equal to the duration of one code bit because the four new paths are present in the shift registers after the second clock pulse has been produced. The second of the two bits which consecutively appear at the $Z_p$ output of the multiplexer 9309 is always the decoded data bit. Since the bit rate of the code is twice that of the data, the duration of the decoded bit must be doubled. This is achieved by means of a latch incorporated in the circuit of the control unit discussed below.

6.3 The control unit

Fig.6.5 shows the circuit diagram of the control unit and Fig.6.6 its timing diagram.

It is necessary for each pair of code bits to be fed from the code input of the decoder to the metric circuit. For this reason the code sequence is delayed for one bit-period by feeding $x_{n1}$ and $x_{n2}$ to the circuit of Fig.6.2 via the latch 9314 (X).

The latch command coincides with the clock pulse for the metric registers. It is thus possible for the calculation of the new metric values and the shift-in and read-out information $a$, $b$, $c$, $d$ and $S_0$, $S_1$ to start at this instant. Once this calculation has been completed, the path registers can be filled.

"PE path" must then obviously be low, and "CP path" is transferred from L to H. Subsequently "PE path" should become high; at the next positive flank of "CP path" the four new paths are thus stored in the shift register. At that instant the read-out information should be available. After reading-out, the decoded bit is available at $Z_d$ of multiplexer 9309. This bit is stored by the latch 9314 (X) until, after two clock pulses, a new bit is available at $Z_d$ of multiplexer 9309.

It will be clear that the speed of the decoder is limited by the time the metric circuit needs for calculating the read-in information of the path. Theoretically (according to the specification of the integrated circuits used), this time is about 100 ns. The maximum repetition frequency of the input clock pulse is determined mainly by this time and the width of the CP metric, which results in a maximum of about 2,5 megabits per second for the data signal.

To minimize the influence of mains and similar interferences on the decoder, all flip-flops and one-shots are controlled by a clock with the largest practicable duty cycle. In addition, various bypass capacitors are incorporated in the circuit. The circuit is nevertheless still affected to some extent by interference. This should be taken into account when carrying out measurements with very small error rates.

6.4 The error detector

By extending the 3-bit shift register of the encoder from 3 to 17 bits, the input signal can be brought roughly into phase with the decoded output signal. The latch 9314 (X) will then bring it completely into phase. If the input and output data sequences are then applied to an exclusive OR-gate it will register a 1 for each error in the decoded sequence. Feedback this output directly to a counter would result in two consecutive errors being recorded as only one, since there would be only one positive flank to actuate the counter. This is avoided by feeding the output signal of the XOR-gate, together with a clock signal produced by a one-shot, to a NAND-gate, which is in turn followed by a NAND-buffer. This results in a separate positive-going pulse for each individual error.
Fig. 6.5 Circuit of the coder, the control unit and the error detector.
Fig. 6.6 Timing of the control unit.
7 MEASUREMENTS

In order to measure the bit error rate as a function of the crossover error rate of a binary symmetric channel, the coder and decoder can be linked by a simulated channel with a variable error rate. The measuring set-up is then as shown in Fig. 7.1.

J. Alma [15] devised a binary symmetric channel simulator in which the crossover probability can be adjusted stepwise in steps of \(2^{-1}\) from \(2^{-2}\) to \(2^{-12}\). To generate the errors, a random data generator is incorporated in the circuit. The only random data generator available at the time of the investigations had to be controlled by a clock frequency not exceeding 50 kHz; the maximum permissible code bit rate was thus limited to 4 kbit/s, so the maximum data bit rate to only 2 kbit/s. Since, according to eq.(5.4), the calculated upper bound of \(P_B\) is \(50 p^2 (1 + 3.2p) / (1 - 16p^2)\), the expected bit error rate for \(p = 2^{-12}\) is at the most \(10^{-9}\). At the maximum data bit rate of 2 kbit/s this amounts to 1 error every 6 days. It will be clear that carrying out error probability measurements in this range would be quite impracticable, the more so because the sensitivity to man-made interference would render the results unreliable. For this reason the measurements were limited to \(p = 2^{-9}\).

The curves \(P_B(p)\) shown in Fig. 7.2 were plotted by feeding the outputs “counter” and “data clock” to a programmable counter-timer. The latter directly computed \(P_B\), i.e. the ratio of the number of bit errors in the decoded data sequence to the total number of data bits. The results were checked by measuring the crossover error rate \(p\) in the same way.

As mentioned in Section 6.2, measurements were carried out not only with a pseudo-random data signal fed to the input of the encoder, but also with shorted input (all-zeros codeword) and with open-circuited input (all-ones codeword). The resulting curves are also plotted in Fig. 7.2.

For small values of \(p\) the measured bit error rate appears to be in excellent agreement with the bound given by eq.(5.4) and closely follows the asymptote \(50 p^2\) (cf. Fig. 5.2).

The bit error probability was also measured as a function of the path register length for three values of \(p\), viz. \(2^{-3}\), \(2^{-5}\) and \(2^{-7}\). These measurements, which are plotted in Fig. 7.3, showed that increasing the path register lengths beyond 12 bits scarcely reduced \(P_B\) any further. This confirms the conclusion of Heller and Jacobs [6], Viterbi [5], Odenwalder [13] and others, according to which a path register with a length of 4 to 5 times the constraint length is sufficient to justify disregarding path memory truncation errors.

The maximum data bit rate at which encoder and decoder still operate reliably without a binary symmetric channel being used, proved to be 2.46 Mbits/s, almost as calculated in Section 6.3.

The total power consumption of the installation was 16 W (3.2 A at 5 V).

It should be recognized that availability of a faster random data generator would have allowed the measurements to be carried out in less time and with greater accuracy.

---

![Fig. 7.1 Block diagram of the measuring circuit.](image-url)
Fig. 7.2 Measured bit error rate $P_B$ as a function of $p$. The dash-dot curve applies to a random input signal, the solid curve to the all-zeros codeword and the dashed curve to the all-ones codeword.

Fig. 7.3 Measured bit error rate as a function of the path register length for three values of $p$. 
8 SUGGESTIONS FOR SIMPLIFYING THE CIRCUIT

8.1 Simplification of the metric circuit

At the time the circuit of Fig.6.1 was built, it was not realized that the four metric values would never exceed 3 (cf. Section 3.3); this explains why the design of the whole circuit was based on 3 bits. However, it can be proved that the metric values, even before subtracting $M_{\text{min}}$, never will exceed 3, so that the part of the circuit which follows the table of handling 2 bits.

Four 9322 multiplexers could be replaced by one 8-bit latch (e.g. 9308); moreover, the greater fan-out of the latter would then render the four buffers of the 7483s superfluous. It can also be proved that $M_{\text{min}}$ can only be either 0 or 1, i.e. 1 bit, so that the entire circuitry for the two's complement could be omitted. This will become clear by considering that the two's complement of 0 is 0 0 0 (for 2 bits), and that of 1 is 1 1 1. It would therefore have been permissible to link both $B_2$ and $B_3$ of the 7483s to $Z_a$ of the 9322 (M) via a buffer.

8.2 Large-scale integration

As explained in Section 3.3 (cf. Table 4), there are only 31 possible combinations of metric values. After every possible combination $M^{(i)}$, $x_n$ will be either 0 0, 0 1, 1 0 or 1 1. $M^I = M^{(i)} - M_{\text{min}}$ will then again be one of the 31 combinations, as illustrated by the following example.

Assume $M^{(i)}$ to be 0, 1, 1, 1 (8th column of Table 4), then

\[ M^{d'} = \begin{cases} 
0 & \text{if } x_n = 0 \\
1 & \text{if } x_n = 1
\end{cases} \]

(cf. Table 3)

In accordance with the 20th column of Table 4;

\[ M^{d''} = \begin{cases} 
1 & \text{if } x_n = 0 \\
0 & \text{if } x_n = 1
\end{cases} \]

(cf. Table 3)

These operations can be performed by a fairly simple combinational network to the inputs of which $x_n$, $M^{d'}$, $M^{d''}$, $M_{c}'$, $M_{c}''$, $M_{d}'$, $M_{d}''$ are applied (for each, 2 bits are required), whilst at the outputs $M_d$, $M_d'$, $M_d''$, $M_{c}'$, $M_{c}''$, $M_{d}'$, $M_{d}''$ (2 bits each) and $a$, $b$, $c$, $d$, $S_0$, $S_1$ (1 bit each) become available. It might be feasible to realize such a circuit on a single LSI chip, and if an 8-bit latch could also be accommodated on the same chip, only 12 connections would have to be made to it (cf. the metric circuit of Fig.6.1 and Figs. 6.2 and 6.3).
Given, for odd values of $k$:

$$P_k = \sum_{e=(k+1)/2}^{k} \binom{k}{e} p^e q^{k-e},$$

and for even values of $k$:

$$P_k = \frac{1}{2} \left( \binom{k}{k/2} p^{k/2} q^{k/2} + \sum_{e=k/2+1}^{k} \binom{k}{e} p^e q^{k-e} \right),$$

or, with $k = 2n-1$ and $k = 2n$ respectively:

$$P_{2n-1} = \sum_{e=n}^{2n-1} \binom{2n-1}{e} p^e q^{2n-1-e},$$

$$P_{2n} = \frac{1}{2} \left( \binom{2n}{n} p^n q^n + \sum_{e=n+1}^{2n} \binom{2n}{e} p^e q^{2n-e} \right).$$

To be proved: $P_k = P_{k-1}$ for even values of $k$, or $P_{2n} = P_{2n-1}$. 

Proof:

$$P_{2n} = \frac{1}{2} \left( \binom{2n}{n} p^n q^n + \sum_{e=n+1}^{2n} \binom{2n}{e} p^e q^{2n-e} + p2n - \binom{2n}{n} p^n q^n \right), \tag{A1.1}$$

whilst

$$\binom{2n}{e} = \frac{(2n)!}{2^n (n-e)! e!} = \frac{(2n-e)!(2n-1)!}{(2n-e)e!} \frac{(2n-1)!}{(2n-e)! e!} = \frac{(2n-1)!}{(2n-e)! e!} \cdot \frac{(2n-1)!}{(2n-e)! (e-1)!} = \binom{2n-1}{e} + \binom{2n-1}{e-1}. \tag{A1.2}$$

Substitution of eq.(A1.2) in eq.(A1.1) yields:

$$P_{2n} = -\frac{1}{2} \left( \binom{2n}{n} p^n q^n + \sum_{e=n+1}^{2n-1} \binom{2n-1}{e} p^e q^{2n-e} + \sum_{e=n}^{2n-1} \binom{2n-1}{e} p^e q^{2n-e} + p2n \right), \tag{A1.3}$$

Substituting $i$ for $e-1$ in the third term gives:

$$P_{2n} = -\frac{1}{2} \left( \binom{2n}{n} p^n q^n + \sum_{i=n}^{2n-1} \binom{2n-1}{i} p^{i+1} q^{2n-1-i} + p2n \right)$$

$$= -\frac{1}{2} \left( \binom{2n}{n} p^n q^n + \sum_{i=n}^{2n-1} \binom{2n-1}{i} p^{i+1} q^{2n-1-i} + \binom{2n-1}{n-1} p^n q^n - p2n + p2n \right). \tag{A1.4}$$
moreover,

\[
\binom{2n-1}{n-1} = \frac{(2n-1)!}{(n-1)!n!} = \frac{(2n)!}{2n\cdot n!n!} = \frac{1}{2} \frac{(2n)!}{n!} = \frac{1}{2} \binom{2n}{n}. \tag{A1.5}
\]

From eqs (A1.4) and (A1.5):

\[P_{2n} = qP_{2n-1} + pP_{2n-1} = P_{2n-1} \tag{A1.6}\]
q.e.d.

Appendix 2

As proved in Appendix 1, \(P_k = P_{k-1}\) for even values of \(k\); in that case

\[P_k = \sum_{e=k/2}^{k-1} \left( \binom{k-1}{k/2 + e} \right) p^e q^{k-e} \leq \sum_{e=k/2}^{k-1} \binom{k-1}{e} q^{k-1} p^e, \tag{A2.1}\]

since

\[\left( \binom{k-1}{k/2 + i} \right) \geq \left( \binom{k-1}{k/2} \right), \quad (i = 1, 2, 3, ..., k/2 - 1). \tag{A2.2}\]

Considering that

\[\binom{k-1}{k/2} = \frac{(k-1)!}{k/2 \cdot (k/2 - 1)!} = \frac{k!}{2 \cdot (k/2 - 1)!} = 4 \frac{k-1}{k} < 4, \tag{A2.3}\]

eq.(A2.1) may be written:

\[P_k < \left( \binom{k'-1}{k'/2} \right)^2 k^{k-k'} q^{k-1} (\frac{p}{q})^{k/2} \frac{1 - (p/q)(k/2)}{1 - p/q}, \tag{A2.4}\]

in which \(k'\) is the minimum even free distance of the convolutional code concerned; in the case at issue \(k' = 6\).

Hence, in eq.(A2.4), \(k' < k\).

Rearrangement of eq.(A2.4) yields

\[P_k < \Gamma (2 \sqrt{p}) k^{k-k'} q^{k-1} (\frac{p}{q})^{k/2} \frac{1 - (p/q)(k/2)}{1 - p/q}, \tag{A2.5}\]

where

\[\Gamma \triangleq \binom{k'-1}{k'/2}^2 \cdot k'.\]

In the case at issue,

\[\Gamma = \left( \frac{5}{3} \right)^2 \cdot 6 = \frac{5}{32}.\]
In the last term of eq.(A2.5) we now make the substitution 
$q = \frac{1}{2} + x$; and, from considerations of symmetry we can, 
without departing from generality, assume that $\frac{1}{2} < q \leq 1$, 
whence $0 < x \leq \frac{1}{2}$. The last term then becomes

\[
\frac{(\frac{1}{2} + x)^{k/2} - (\frac{1}{2} - x)^{k/2}}{2x} = A(k) + B(k)x^2 + C(k)x^4 + \ldots ,
\]  
(A2.6)

where $A$, $B$, $C$ are non-negative constants which depend 
exclusively on $k$. Eq.(A2.6) increases monotonically for 
$x > 0$ and hence reaches a maximum at the maximum value 
of $x$. Substituting $\frac{1}{2}$ for $x$ yields 1, so that

\[
\frac{q^{k/2} - p^{k/2}}{q - p} \leq 1.
\]  
(A2.7)

Then

\[ P_k < \Gamma (2 \sqrt{p})^k \]  
(A2.8)

for even values of $k$.

Appendix 3

In general (cf. Viterbi [5]), if $P_k < F(p) ^k$:

\[
P_E < T(D) \quad \left| D = F(p) \right. \]  
(A3.1)

\[
P_B < \frac{dT(D,N)}{dN} \quad \left| N = 1, D = F(p) \right. \]  
(A3.2)

In general $T(D)$ may be expressed by

\[ T(D) = a_1 D + a_2 D^2 + a_3 D^3 + \ldots , \]  
(A3.3)

and then in general:

\[ P_E < a_1 P_1 + a_2 P_2 + a_3 P_3 + \ldots , \]  
(A3.4)

or, making use of the fact that $P_1 = P_2, P_3 = P_4, \ldots$ as proved in Appendix 1,

\[ P_E < (a_1 + a_2)P_2 + (a_3 + a_4)P_4 + \ldots . \]  
(A3.5)

Making use of eq.(A2.8) and the definitions of $\Gamma$ and $k'$ given in Appendix 2, we may write

\[ P_E < \Gamma \left\{ (a_1 + a_2) (2 \sqrt{p})^2 + (a_3 + a_4) (2 \sqrt{p})^4 + \ldots . \right\} \]  
(A3.6)

To analyse this expression, we shall try to find a function of $T(D)$ which assumes the general form

\[ G \{ T(D) \} = (a_1 + a_2)D^2 + (a_3 + a_4)D^4 + \ldots , \]  
(A3.7)

which is satisfied by

\[ \frac{T(D) + T(-D)}{2} + D \frac{T(D) - T(-D)}{2} . \]  
(A3.8)
Eq. (A3.6) may thus be expressed by

\[
P_E < \Gamma \left[ \frac{T(D) + T(-D)}{2} + D \frac{T(D) - T(-D)}{2} \right] \quad D = 2 \sqrt{p}.
\]  
(A3.9)

In an analogous way it can be shown that

\[
P_B < \Gamma \left[ \frac{dT(D, N) + dT(-D, N)}{dN} + D \frac{dT(D, N) - dT(-D, N)}{dN} \right] \quad N = 1, D = 2 \sqrt{p}.
\]  
(A3.10)

These upper bounds are thus generally valid for any convolutional code. We shall now show that eqs. (5.3) and (5.4) are indeed obtained by substituting in eqs. (A3.9) and (A3.10):

\[
T(D) = \frac{D^5}{1 - 2D} \quad T(D, N) = \frac{D^5 N}{1 - 2D} \quad \text{and} \quad \Gamma = \frac{5}{32}.
\]

Eq. (A3.9) then yields

\[
P_E < \frac{5}{32} \left| \frac{D^5}{1 - 2D} - \frac{D^5}{1 + 2D} + \frac{D^6}{1 - 2D} + \frac{D^6}{1 + 2D} \right| D = 2 \sqrt{p}
\]

\[
= \frac{5}{32} \left| \frac{D^5 + 2D^6 - D^5 + 2D^6 + D^6 + 2D^7 + D^6 - 2D^7}{2(1 - 4D^2)} \right| D = 2 \sqrt{p}
\]

\[
= \frac{5}{32} \left| \frac{3D^6}{1 - 4D^2} \right| D = 2 \sqrt{p} = \frac{5}{32} \cdot \frac{3 \times 64p^3}{1 - 4 \times 4p} = \frac{30p^3}{1 - 16p},
\]  
(5.3)

and eq. (A3.10) becomes:

\[
\frac{dT(D, N)}{dN} \bigg|_{N = 1} = \frac{D^5}{(1 - 2D)^2}, \quad \frac{dT(-D, N)}{dN} \bigg|_{N = 1} = \frac{-D^5}{(1 + 2D)^2}
\]

\[
P_B < \frac{5}{32} \left| \frac{D^5}{(1 - 2D)^2} - \frac{D^5}{(1 + 2D)^2} + \frac{D^6}{(1 - 2D)^2} + \frac{D^6}{(1 + 2D)^2} \right| D = 2 \sqrt{p}
\]

\[
= \frac{5}{32} \left| \frac{D^5(1 + 2D)^2 - D^5(1 - 2D)^2 + D^6(1 + 2D)^2 + D^6(1 - 2D)^2}{2(1 - 4D^2)^2} \right| D = 2 \sqrt{p}
\]

\[
= \frac{5}{32} \left| \frac{5D^6 + 4D^8}{(1 - 4D^2)^2} \right| D = 2 \sqrt{p} = \frac{50p^3}{(1 - 16p)^3},
\]  
(5.4)
Bibliography


Acknowledgement

The author is greatly indebted to his thesis advisors Prof. J.P.M. Schalkwijk and A.P. Verlijdonk of the Department of Electrical Engineering at the Eindhoven University of Technology, and to K.A. Post of the Mathematics Department, for helpful discussions and suggestions in preparing this report.
EINDHOVEN UNIVERSITY OF TECHNOLOGY
THE NETHERLANDS
DEPARTMENT OF ELECTRICAL ENGINEERING

Reports:

1) Dijk, J., M. Jeuken and E.J. Maanders
AN ANTENNA FOR A SATELLITE COMMUNICATION GROUND STATION
ISBN 90 6144 001 7

2) Veefkind, A., J.H. Blom and L.Th. Rietjens
THEORETICAL AND EXPERIMENTAL INVESTIGATION OF A NON-EQUILIBRIUM
to the Symposium on a Magnetohydrodynamic Electrical Power

3) Boom, A.J.W. van den and J.H.A.M. Melis
A COMPARISON OF SOME PROCESS PARAMETER ESTIMATING SCHEMES.
TH-report 68-E-03. September 1968. ISBN 90 6144 003 3

4) Eykhoff, P., P.J.M. Ophey, J. Severs and J.O.M. Oome
AN ELECTROLYTIC TANK FOR INSTRUCTIONAL PURPOSES REPRESENTING THE
ISBN 90 6144 004 1

5) Vermij, L. and J.E. Daalder
ENERGY BALANCE OF FUSING SILVER WIRES SURROUNDED BY AIR.

6) Houben, J.W.M.A. and P. Massee
MHD POWER CONVERSION EMPLOYING LIQUID METALS. TH-report 69-E-06.
February 1969. ISBN 90 6144 006 8

7) Heuvel, W.M.C. van den and W.F.J. Kersten
VOLTAGE MEASUREMENT IN CURRENT ZERO INVESTIGATIONS. TH-report 69-E-07.
September 1969. ISBN 90 6144 007 6

8) Vermij, L.
ISBN 90 6144 008 4

9) Westenberg, J.Z.
SOME IDENTIFICATION SCHEMES FOR NON-LINEAR NOISY PROCESSES.

10) Koop, H.E.M., J. Dijk and E.J. Maanders
ISBN 90 6144 010 6

11) Veefkind, A.
NON-EQUILIBRIUM PHENOMENA IN A DISC-SHAPED MAGNETOHYDRODYNAMIC


13) Teuling, D.J.A.
ELECTRONIC IMAGE MOTION COMPENSATION IN A PORTABLE TELEVISION CAMERA.
14) Lorencin, M.

15) Smets, A.J.

16) White, Jr., R.C.

17) Talmon, J.L.

18) Kalásek, V.

19) Hosselet, L.M.L.F.

20) Arts, M.G.J.

21) Roer, Th.G. van de

22) Jeukens, P.J., C. Huber and C.E. Mulders

23) Dijk, J. and E.J. Maanders

24) Kregting, J. and R.C. White, Jr.


26) Brenner, H.

27) Bokhoven, W.M.G. van

28) Boeschoten, F.

30) Kessel, C.G.M. van and J.W.M.A. Houben

31) Veeffkind, A.

32) Daalder, J.E. and C.W.M. Vos

33) Daalder, J.E.

34) Huber, C.

35) Bastian, C. et al.

36) Blom, J.A.

37) Lier, M.C. van and R.H.J.M. Otten

38) Andriessen, F.J., W. Boerman and I.F.E.M. Holtz


40) Goes, W.P.

41) Damen, A.A.H.
42) Dijk, G.H.M. van
THEORY OF GYRO WITH ROTATING GIMBAL AND FLEXTURAL PRIOTS.

43) Breimer, A.J.
ON THE IDENTIFICATION OF CONTINUOUS LINEAR PROCESSES. TH-report

44) Lier, M.C. van and R.H.J.M. Otten
ISBN 90 6144 044 0

45) Bastian, C. et al.

46) Roer, Th.G. van de

47) Leliveld, W.H.

48) Damen, A.A.H.

49) Meeberg, L. van de

50) Poel, A.P.M. van der

51) Sampic, G.

52) Schalkwijk, J.P.M.
CODING FOR A COMPUTER NETWORK. TH-report 74-E-52. October 1974. ISBN 90 6144 052 1

53) Stapper, M.

54) Schalkwijk, J.P.M. and A.J. Vinck

55) Yakimov, A.