Symbol synchronization in convolutionally coded systems
Baumert, L.; McEliece, R.J.; van Tilborg, H.C.A.

Published in:
IEEE Transactions on Information Theory

DOI:
10.1109/TIT.1979.1056044

Published: 01/01/1979

Citation for published version (APA):
A t-error-correcting code is perfect if the covering radius is t. The code is quasi-perfect if the covering radius is t + 1.

Let \( \beta \) be an element of order \( n = 2^m - 1 \). The largest cyclic code whose generator polynomial \( g(x) \in \mathbb{GF}(2)[x] \) has the zeros \( \beta, \beta^2, \ldots, \beta^{t-1} \) but not \( \beta^t \) is defined to be a primitive BCH code of designed distance \( d \) and is here denoted by \( B(d) \). Note that \( d \) must be odd if \( B(d) \) exists.

The code \( B(3) \) is the Hamming code, which is a one-error-correcting perfect code. Gorestein, Peterson, and Zierler [1] proved that \( B(5) \) is a two-error-correcting quasi-perfect code. They also proved that \( B(15) \) is a three-error-correcting code which has covering radius at least five, and thus \( B(15) \) is not quasi-perfect. Later Van der Horst and Berger [2], Assmus and Mattson [3], and Helleseth [4] proved that \( B(7) \) has covering radius five.

In this correspondence we will prove a conjecture due to Gorestein, Peterson, and Zierler [1], which says that \( B(d) \) is never quasi-perfect when \( d > 7 \).

For the proof we will need the following lemmas.

Lemma 1: If \( d = 2^r - 1, r < m \), then \( B(d) \) exists and has actual minimum distance \( d \).

Lemma 2: If \( d = 2^r - 2^s - 1, \) then \( B(d) \) exists and has actual minimum distance \( d \).

Lemma 1 is theorem 9.4 in Peterson and Weldon [6]. Lemma 2 is proved by Kasami and Lin [7].

Theorem 1: No primitive binary t-error-correcting BCH code is quasi-perfect when \( t > 2 \).

Before proving Theorem 1 we prove the following stronger result.

Theorem 2: Let \( \rho_d \) and \( \tau_d \) denote the covering radius and actual error correcting ability of \( B(d) \), respectively, and let \( 3 < r < m - 1 \).

i) If \( 2^r - 2^r - 1 < d < 2^{r+1} - 1 \) where \( s \) is one of the numbers \( \frac{r}{2}, \frac{r}{2} + 1, \ldots, r - 2 \), then

\[
\rho_d - \tau_d > \frac{2^{r} - 3}{2^{r+1} - 1} (\tau_d + 1).
\]

ii) If \( 2^{r+1} - 1 < d < 2^{r+1} - 1 \), then

\[
\rho_d - \tau_d > \frac{2}{2^{(r+1)/2}} - 1 (\tau_d + 1).
\]

Proof:

i) Let \( 2^{r+1} - 1 < d < 2^{r+1} - 1 \) for some \( s = \frac{r+1}{2}, \frac{r+1}{2} + 1, \ldots, r - 2 \), where \( 3 < r < m - 1 \). By Lemma 2, \( B(2^{r+1} - 1) \) and \( B(2^{r+2} - 1) \) exist, and we have

\[
B(2^{r+1} - 1) \subset B(d) \subset B(2^{r+2} - 1).
\]

Since \( B(d) \subset B(2^{r+2} - 1) \), we can choose \( \alpha \in B(2^{r+2} - 1) \) \( \cdot B(d) \). Here \( \alpha \) has distance at least \( 2^{r+2} - 1 \) from every element in \( B(d) \). From the definition of the covering radius it follows that

\[
\rho_d > 2^r - 2^{r+1} - 1.
\]

Since \( B(2^{r+2} - 1) \subset B(d) \), we get by Lemma 2

\[
\rho_d - \tau_d > 2^{r+1} - 1 - 2^{r} - 1.
\]

Combining (1) and (2) we have

\[
\rho_d - \tau_d > 2^{r} - 1 (2^{-r} - 3)
\]

which combined with (2) gives

\[
\rho_d - \tau_d > (\tau_d + 1)(2^{-r} - 3)/(2^{-r} - 1).
\]

This proves i).

u) This is proved using the same method as in the proof of i).

Proof of Theorem 1: Since the only \( B(d) \) with \( d > 2^{m-1} - 1 \) is the perfect binary repetition code \( B(2^{m-1}) \), it is sufficient to prove that \( \rho_d - \tau_d > 1 \) when \( 5 < d < 2^{m-1} - 1 \).

Let \( 5 < d < 2^{m-1} - 1 \). We can choose \( r \) such that \( 3 < r < m - 1 \) and \( 2^{r+1} - 1 < d < 2^{r} - 1 \). Further \( d \) belongs to one of the two cases i) or ii) of Theorem 2.

Note that we have

\[
\rho_d - \tau_d > \frac{1}{2} (\tau_d + 1), \quad \text{when } d \text{ belongs to case i)}
\]

\[
\rho_d - \tau_d > \frac{1}{2} (\tau_d + 1), \quad \text{when } d \text{ belongs to case ii)}
\]

Hence we always have \( \rho_d - \tau_d > 1 \) since \( \tau_d > 3 \), and therefore \( B(d) \) is not quasi-perfect except when \( d = 5 \).

From the proof above we get the following corollary.

Corollary: If \( \tau_d > 2 \) and \( \tau_d \neq 2^{r} - 2^{r+1} - 1 \), then \( \rho_d - \tau_d \geq \frac{1}{2} (\tau_d + 1) \).

REFERENCES


Symbol Synchronization in Convolutionally Coded Systems

LEONARD D. BAUMERT, ROBERT J. McELIECE, MEMBER, IEEE, AND HENK C. A. VAN TILBORG

Abstract—Alternate symbol inversion is sometimes applied to the output of convolutional encoders to guarantee sufficient richness of symbol transition for the receiver symbol synchronizer. A bound is given for the length of the transition-free symbol stream in such systems, and those convolutional codes are characterized in which arbitrarily long transition free runs occur.

I. INTRODUCTION

Many digital communication systems derive symbol synchronization from the transitions in the received symbol stream. In such systems unusually long sequences of all zeros or all ones can cause temporary loss of synchronization and thus data loss. To avoid this problem, alternate symbols of the data stream are inverted; presumably a long alternating string is less likely than a long constant string.

Suppose the symbol stream is the alternately inverted output of a convolutional encoder. How long a constant string occurs?
Theorem 1: Let C be an \((n,k)\) convolutional code over \(GF(2)\) with generator matrix \(G\). Then a codeword can be found with an infinite run of alternating symbols if and only if there exists a linear combination \(u = \sum_{i=1}^{k} u_i g_i\) of the rows \(g_i\) of \(G\) such that
\[
[0,1,...,0,1] \text{ or } [1,0,...,1,0] \mod (1 + D, n \text{ even })\]

or
\[
[1,D,\ldots,D,1] \text{ or } [D,1,\ldots,1,D] \mod (1 + D^2, n \text{ odd })\]

Proof (Sufficiency): When \(n\) is even, consider the codeword produced by the inputs \(a_i/(1 + D)\) applied to the rows \(g_i\), where \(v = u g_i\). Note that this same codeword is produced by applying \(1/(1 + D) (= 1111 \cdots)\) to each row of the equivalent encoder whose rows are \(g_i\). Thus after an initial transient the output will be \(v_1(1), \ldots, v_n(1)\) and since \(v_1(1) = v_0(1)\) modulo \(1 + D\) the result follows. For \(n\) odd note that \(u/D(1 + D^2)\) modulo \(1 + D^3\) means that the sum of its even coefficients is 0 and the sum of its odd coefficients is 1. If this is done, then by time \(t = 0\), where \(t = (\text{max} \text{ (deg } f_i))\) it is sufficient to check the initial results of this codeword of \(C\) with an infinite alternating run.

Proof (Necessity): When \(n\) is even an infinite run of alternating symbols results from the juxtaposition of \(n\)-tuples of the form \(10\cdots 10\). For definiteness, assume the former occurs. Then, if a codeword of \(C\) contains such an infinite run, there exists a codeword \(u\) such that
\[
u = b + \frac{D^2}{1 + D^2} [1,0,...,1,0]
\]
where \(b\) is an \(n\)-tuple of polynomials (of degrees \(\leq s\)) which describes the initial segment of \(u\). Let \(v(D) = (1 + D) u(D)\). Obviously, \(v(D)\) is polynomial and \(v(D) = [1,0,...,1,0] \mod (1 + D^2)\). Similarly, for \(n\) odd, \(C\) contains
\[
\begin{align*}
\omega &= h' + \frac{D^2}{1 + D^2} [1,D,\ldots,D,1].
\end{align*}
\]
Define \(v(D)\) as \((1 + D^2)\omega(D)\). It follows as above that
\[
v(D) = D^2 [1,D,\ldots,D,1] \mod (1 + D^2)
\]
and the proof is complete.

If a basic encoder \(G\) is known for \(C\) then only \(2^k\) (respectively, \(4^k\)) linear combinations \(v = \sum a g_i\) need be tried, for then the \(a_i\) can be restricted to 0,1 (respectively, 0,1,\(D,1 + D\)) when \(n\) is even (respectively, \(n\) is odd). Even more efficiently, a row reduction could be used to determine whether or not the required vector was in the row space of \(G\) modulo \(1 + D\) (or \(1 + D^2\)).

The case \(k = 1\) is particularly important. Here, basic just means that the \(n\) polynomials making up the single generator \(g_1\) have no common polynomial divisor and the test amounts to reducing \(g_1(1 + D)\) modulo \(1 + D^2\).

It is also possible to test for the presence of an infinite alternating run in terms of the dual code (see Corollary to Theorem 2 below)

Theorem 2: Suppose an \((n,n-1)\) convolutional code \(C\) over \(GF(2)\) is given and \(f = [f_1,\ldots,f_n]\) generates the dual code, where \(gcd (f_1,\ldots,f_n) = 1\). Then there is an infinite run of alternating symbols in some codeword of \(C\) if and only if

- \((n\text{ even}) \quad \Sigma f_{i+\alpha} \equiv 0 \mod (1 + D)\) for \(\alpha = 0\) or \(\alpha = 1\)
- \((n\text{ odd}) \quad \Sigma f_{i+D} \equiv 0 \mod (1 + D^2)\).

Proof: Since \((f_1,\ldots,f_n)\) is a set of codewords of the dual code \(C\) with alternating symbols, we can write the codewords produced by the inputs \(\alpha_i/(1 + D)\) applied to the rows \(g_i\) of \(G\), where \(g_i(1 + D)^n = \sum a_i g_i\). Thus after an initial transient the output will be \(u_1(1), \ldots, u_n(1)\) and since \(u_1(1) = v(1)\mod (1 + D^2)\), the result follows. For \(n\) odd note that \(u(D) = D^2 v(D)\) modulo \(1 + D^3\) means that the sum of its even coefficients is 0 and the sum of its odd coefficients is 1. If this is done, then by time \(t = 0\), where \(t = (\text{max} \text{ (deg } f_i))\) it is sufficient to check the inner products of this codeword of \(C\) with an infinite alternating run.

In both cases the necessity of the above conditions is immediate. (For \(n\) odd the coefficients referred to are \(a,b\) from \(\Sigma f_{i+D} \equiv (aD + b) \mod (1 + D^2)\).

On the other hand, the above conditions obviously guarantee the existence of a codeword \((\cdots \cdot 1010 \cdot 10 \cdots)\) extending infinitely in both directions. However, only codewords "starting" at some finite time are of concern, and it remains to be shown that such a codeword is in the code. But this is trivial; it amounts to using the same input sequences truncated to start at some time \(t_0 = \cdots = t_{q-1} = 0\) for \(t < t_0\). If this is done, then by time \(t_0 + t + \delta\), where \(\delta\) is the overall constraint length, the coders shift registers will be set exactly as they were when generating the doubly infinite sequence. Thus from \(t_0 + \delta\) on the output will be an infinite alternating run.

Suppose an \((n,k)\) convolutional code \(C\) over \(GF(2)\) with generator matrix \(F\) for its dual code is given. Suppose \(F\) is a basic encoder, i.e., the gcd of \(n - k\) by \(n - k\) subdeterminants is 1, then \(f = [f_1,\ldots,f_n]\) in any row of \(F\) it follows that \(f = 1\).

Let \(C_i (i = 1,\ldots,n-k)\) be the \((n,n-1)\) convolutional code dual to the \(i\)th row of \(F\). Clearly,
\[
C = \bigcap_{i=1}^{n-k} C_i
\]
and the maximum length of alternating runs in some codeword of \(C\) has length \(l = l(C) < \min l(C_i)\).

Corollary: When \(n\) is odd, \((n,k)\) convolutional code \(C\) over \(GF(2)\) contains a codeword with an infinite run of alternating
symbols if and only if every row of a basic generator matrix $F$ for $C$ satisfies the congruences of Theorem 2. When $n$ is even it is further necessary that this be true for the same value of $a$ (0 or 1).

**Note:** Suppose $n$ is even and $L(C_i)=L(C_j)=\infty$ with $a\neq 1$ for $C_i$ and $a\neq 0$ for $C_j$. Add row $j$ to row $i$ in $F$; this gives an equivalent basic encoder which has $L(C_i)<\infty$.

### III. Bounds for Finite Runs of Alternating Symbols

If no codeword contains an infinite run of alternating symbols the question arises as to the maximum length $L$ of such a finite run. It is easy to give a bound for $L$ in terms of the generators for the dual code. From this bound it is possible to derive another bound (in general, weaker) which has the advantage that it can be applied directly without knowledge of the dual (see the Corollary to Theorem 3, below). In Section IV these bounds are applied to some specific examples.

Suppose $[f_1, \ldots, f_n]$ is a generator matrix for an $(n, 1)$ convolutional code $C$ over $(\mathbb{F}_2)^d$ with $d=\deg (\deg f_j)$. Then

$$f_{10} f_{30} \cdots f_{a_0} f_{13} \cdots f_{a_1} f_{as} \cdots f_{as},$$

is its associated bit pattern. Let $s$ be the number of symbols occurring between the first and last nonzero symbols $f_1$ inclusively. If $(f_{1}, \ldots, f_{s})=1$, $s$ is the minimum length of any nonzero codeword of $C$ and

$$n(d-1)+2 \leq s \leq n(d+1).$$

**Theorem 3:** Let $C$ be an $(n, n-1)$ convolutional code over $(\mathbb{F}_2)^2$ with generator matrix $G$ for its dual code given by $[f_1, \ldots, f_n]$ where $(f_1, \ldots, f_n)=1$. Suppose no codeword of $C$ contains an infinite run of alternating symbols. Then the maximum run of alternating symbols in any codeword of $C$ has length $L=s+n-2$, when $n$ is even or when $n$ is odd and $h(D)=0$ modulo $1+D^2$. If $n$ is odd and $h(D)=1$ or $D$ modulo $1+D^2$, the maximum run of alternating symbols has length $L=s+2n-2$.

Combining this with the limits given above for $s$ yields

$$n(d-1)+2 \leq s \leq n(d+1),$$

**Proof:** Suppose $n$ is even. Then, from Theorem 2, $\Sigma f_{j} \equiv f_{j+1} \equiv 1$ modulo $1+D$. If there were an alternating run of length $s+n-1$ it would have $s$ consecutive symbols which would have inner product zero with the bit pattern of the $f_j$. This contradicts $\Sigma f_{j} \equiv f_{j+1} = 1$, so $L \leq s+n-2$. On the other hand, consider an alternating run of length $s+n$. Change the first and last of these $s$ symbols; the inner products will be correct provided that they match up with the symbols $1, \ldots, s$ and $n+1, \ldots, n+s$. Clearly, this run can be extended to the right and the left to form a codeword of $C$; it is merely a matter of selecting symbols $1 \pm jn$ so that the inner products are zero. Such a codeword could conceivably extend infinitely in both directions; however, using an argument similar to that at the end of Theorem 2, it follows that there is a finite codeword with an alternating run of this length.

If $n$ is odd then, from Theorem 2, $h(D)\equiv 0$ modulo $1+D^2$. If $h(D)\equiv 1+D$ the proof above applies, so $L=s+n-2$. If $h(D)\equiv 1$ or $D$ then one of the inner products is zero but the other is not (see the display shown in the proof of Theorem 2). If there were a run of length $s+n-2$ there would have to be a run of $s$ consecutive symbols where the inner product was zero. On one side or the other of these $s$ symbols would have to be $n$ more symbols from the alternating run of size $s+n-1$. These $n$ symbols together with $s-n$ of the original $s$ symbols would also have to have inner product zero contrary to the hypothesis.

So $L \leq s+2n-2$. As above, a finite codeword of $C$ can be constructed containing an alternating run of length $L=s+2n-2$. It is merely necessary that positions $n+1, \ldots, n+s$ of this run have inner product zero with the bit pattern of the $f_j$.

Recall from the previous section the codes $C_j$, $(n,n-1)$ convolutional codes dual to the rows of $F$, where $F$ was a basic generator matrix for $C$. By and the obvious property

$$C= \bigcap_{i=1}^{n-k} C_i$$

from which it follows that the maximum run of alternating symbols in any codeword of $C$ has length $L=L(C_i)<\min L(C_i)$. Suppose $L(C_i)$ is finite for at least one value of $i$. Then, if $d$ is the maximum degree of any element in the $i$th row of $F$, it follows that

$$L(C)<L(C_i)<\begin{cases} n(d+2)-2, & n \text{ even} \\ n(d+3)-2, & n \text{ odd}. \end{cases}$$

**Corollary:** Suppose an $(n, k)$ convolutional code $C$ over $(\mathbb{F}_2)^d$ is given with basic generator matrix $G$. Let $\mu$ be the maximum degree of the $k \times k$ subdeterminants of $G$. Then either $L(L(C))=\infty$ or

$$L(L(C_i)<\begin{cases} n(\mu+2)-2, & n \text{ even} \\ n(\mu+3)-2, & n \text{ odd}. \end{cases}$$

**Proof:** Under these conditions $C_\perp$ has a generator matrix $F$ (a so-called minimal encoder for $C_\perp$) all of whose entries are of degree $<\mu$. Thus the result follows immediately except when $n$ is even and $L(C_i)=\infty$ for $i=1, \ldots, n-k$. Here if $L$ is finite, a finite bound for it can be determined by replacing row $i$ of $F$ in turn by the sum of row $i$ and row $j$, for $j=1, \ldots, n-k$ ($j \neq i$). Of course in general all this work will not be required but the point is that such transformations do not increase the maximum degree of the elements of the dual encoder and so the bound given above is valid here also.

### IV. Examples

Consider the $(3,2)$ code $C$ generated by the encoder $G$:

$$\begin{bmatrix} D^3+D & D^3+D+1 \\ D^2 & D^3+D+1 \\ 1 & D \end{bmatrix} \pmod{1+D^2}.$$
general. Consider a basic \((4,2)\) convolutional code whose generator matrix modulo \(1 + D\) is
\[
G = \begin{bmatrix}
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & p & q \\
0 & q & p \\
\end{bmatrix}
\]
where \(I\) is an identity matrix or order \(k - 1\) and \(0'\) is a \(k - 1\) by \(k - 1\) matrix of zeros.

Here \(p = p(D) = 1 + D + D^2\) and, for \(n\) even, \(q = q(D) = 1 + D^2 + D^4 + D^n (\mu > 2)\) while for \(n\) odd \(q(D) = 1 + D^3 + D^5 + D^n (\mu > 2)\). \(G\) is obviously basic and minimal. Further Theorem 1 guarantees that no codeword generated by \(G\) contains an infinite run of alternating symbols. That \(G\) generates a codeword with a run of alternating symbols of length \(n\mu + k + 1\) which compares reasonably well with the bounds given by the Corollary. Consider the \((n,k)\) convolutional encoder

Every permutation of the columns of this matrix yields a matrix whose row space contains \([1,0,1,0]\) or \([0,1,0,1]\).

ACKNOWLEDGMENT

The authors wish to thank M. K. Simon and J. G. Smith for bringing this problem to their attention and for suggesting several possible approaches.

REFERENCES


A Note on Optimal Quantization

JAMES A. BUCKLEW AND NEAL C. GALLAGHER, JR., MEMBER, IEEE

Abstract—For a general class of optimal quantizers the variance of the output is less than that of the input. Also the mean value is preserved by the quantization operation.

I. INTRODUCTION

J. Max [1] is generally credited with being the first to consider the problem of designing a quantizer to minimize a distortion measure given that the input statistics are known. Max derives necessary conditions for minimizing the mean square quantization error. These results are summarized in the following equations:

\[
y_j = \int_{x_{j-1}}^{x_j} x f(x) \, dx / P(x_{j-1} < x < x_j)
\]

where \(f(x)\) is the probability density of the variable to be quantized and \(P(x_{j-1} < x < x_j)\) is the probability that \(x\) lies in the interval \((x_{j-1}, x_j)\). The \(y_j\) are output levels and the \(x_{j-1}\) are break points where an input value between \(x_{j-1}\) and \(x_j\) is quantized to \(y_j\). Fleisher [2] later gave a sufficient condition for Max's equations to be the optimal set.

Typically, the above equations are intractable except for simple input densities, causing some researchers to derive approximate formulae for some common densities. Roe [3] derives an approximation for the input interval endpoints assuming that the widths of these intervals are small, i.e., the number of output levels is large. Wood [4] derives a result which states, in effect, that the variance of the output of a minimum mean-square error quantizer should be less than the input variance. He also states that the significance of his result is that the signal and noise are dependent and that no pseudo-independence of the sort considered by Widrow [4] is possible.

However, Wood's derivation assumes the input density to be five times differentiable and that the quantizer input intervals be very small in order to truncate various Taylor series expansions. Furthermore, the derived expression for the output variance is dependent upon the input interval lengths and the input probability density function evaluated at the midpoints of these intervals.

In this note we derive a generalization of Wood's results that eliminates a number of his approximations and generalizes the results to apply to more than just Max quantizers.