BELLCOMM, INC.
955 L'ENFANT PLAZA NORTH, S.W. WASHINGTON, D.C. 20024

COVER SHEET FOR TECHNICAL MEMORANDUM

TITLE- Error-Correcting Coding Techniques For Space Communication Systems

TM- 70-2034-1

DATE- January 21, 1970

FILING CASE NO(S)- 900

AUTHOR(S)- B. P. Tunstall

FILING SUBJECT(S)- (ASSIGNED BY AUTHOR(S)- Error-Correcting Coded Systems

ABSTRACT

Coded communication systems are surveyed and their applicability in space is examined. Theoretical results concerning maximum likelihood detection, channel capacity, probability of error, and code reliability functions are reviewed. Orthogonal, biorthogonal, transorthogonal, BCH, and convolutional codes are discussed in conjunction with maximum likelihood, threshold, Fano, Jelinek, Viterbi, and Ng decoders. Performance curves for various encoder-decoder combinations as determined analytically and by computer simulation are given.

Coded systems are shown to be potentially useful at earth orbital, lunar, or deep space distances on manned or unmanned missions. The use of coding in the IMP satellites, the Pioneer space probes, and the Mariner space probes is summarized. Coding gains of up to 5.5 dB have been achieved in these programs.

SEE REVERSE SIDE FOR DISTRIBUTION LIST

BELLCOMM, INC.
955 L'ENFANT PLAZA NORTH, S.W. WASHINGTON, D. C. 20024

SUBJECT: Error-Correcting Coding Techniques For Space Communication Systems Case 900

DATE: January 21, 1970

FROM: B. P. Tunstall

TECHNICAL MEMORANDUM

I. INTRODUCTION

The purpose of this memorandum is to survey the techniques and performance of error correcting coding and to determine its usefulness in space communication links.

A basic problem facing the designer of a space communication system is the attainment of information and error rate specifications under a weight (and, therefore, power) constraint. Coding offers one tool for helping to solve this problem.

This memorandum discusses the classes of error-correcting codes, channel capacity, code reliability functions and coding techniques. Performance curves determined by computer simulations are given and some present applications are mentioned.

Weight and complexity limitations prevent the use of elaborate decoders in spacecraft; Earth-to-spacecraft links (or uplinks) usually depend on ground based high powered transmitters and high gain antennas to overcome the effect of noise. The constraints on the downlink are just the opposite. The spacecraft transmitter must be low powered and elaborate decoders can be implemented on the ground. Thus, the discussion in this memorandum is concerned with the downlink.

A framework for the discussion to follow is given by the block diagram in Figure 1-1. This coded system is general enough to represent most of the space vehicle systems presently under development or in use.

The data presented to the encoder aboard the spacecraft is assumed to be in binary form. The encoder generates another binary sequence which enters a J stage shift register. The shift register has M possible outputs with $M=2^J$. For each

of the M possible shift register outputs, a waveform $\phi_i(t)$, is generated by the waveform generator. It has been found that the disturbance in a space communication link is primarily receiver thermal noise.[1] Therefore, n(t) is chosen to be stationary white Gaussian noise.

The optimum detector for additive white Gaussian noise is a matched filter or correlation detector operating coherently.[1] The detector shown in Figure 1-1 is also assumed to include quantization and ranking of the matched filter outputs. The correlation interval and the number of quantization levels are discussed in later sections.

A particular system might not employ all of the components shown in Figure 1-1. For example, the coder might be an identity transformation, or M might equal two in which case there would be no shift register.

Feedback coding schemes are not particularly useful beyond the near earth vicinity because of the round trip signal delay, and are not included in this report.

Some of the codes introduced can be used for error detection as well as forward error correction; however, this memorandum focuses on correcting techniques.

In Section II, error-correcting codes are introduced by a simple example which employs maximum likelihood decoding. Codes are broken into two classes, block codes and convolutional codes. Section III considers two theoretical bounds on the performance of codes, the channel capacity and the reliability function. The channel capacity determines how rapidly information can be transmitted, and, roughly speaking, the reliability function shows how rapidly the probability of error decreases as code complexity increases. Section IV discusses specific encoding and decoding techniques and the results of simulating these techniques on a computer. Section V contains a review of the use of coding by NASA.

II. ERROR CORRECTING CODES

A Simple Coding Example

Consider the simplified communication system shown in Figure 2-1. Assume the message, m, is drawn at random from

a finite set, $\left[m_k | k = 1, \cdots, N\right]$, with probability $P\left[m_i\right]$, $i = 1, \cdots, N$, and that the transmitter generates the scalar value, s_k, when the message is m_k. The channel is described by the conditional probability density functions $\left[p_r\ (\rho | s = s_i)\right]$ where ρ is a particular value of the received scalar, r.

Which particular message, m_k, should $\hat{m}$, the receiver output, be set equal to? The answer depends on the criterion of optimality; a common criterion is error probability, and in this report an optimum receiver is defined as one which minimizes probability of error. Assume the optimum receiver sets $\hat{m} = m_k$. Then the conditional probability of correct reception, given that $r = \rho$, is just

$$P\left[C | r = \rho\right] = P\left[m_k | r = \rho\right], \tag{2-1}$$

the conditional probability that the message m_k was sent, given that ρ was received. The unconditional probability of correct reception is

$$P\left[C\right] = \int_{-\infty}^{\infty} P\left[C | r=\rho\right]\ p_r\ (\rho)\ d\rho \tag{2-2}$$

where $p_r(\rho)$ is the unconditional probability density of r. Since $p_r(\rho) \geq o$, the probability of correct reception is maximized (and the probability of error is minimized) by maximizing $P\left[C | r = \rho\right]$ for each ρ. That is, the optimum receiver chooses m_k such that

$$P\left[m_k | r = \rho\right] \geq P\left[m_i | r = \rho\right], \quad i = 1, \cdots, N, \ i \neq k \tag{2-3}$$

whenever ρ is received. The optimum receiver is thus a maximum a posteriori probability receiver.

From a knowledge of the channel, the source, and the received signal, how is m_k chosen? By Bayes rule,

$$P\left[m_i | r = \rho\right] = \frac{P\left[m_i\right]\ p_r\ \left(\rho | m_i\right)}{p_r\ \left(\rho\right)} \tag{2-4}$$

Since $p_r\ \left(\rho\right)$ is independent of receiver structure, i.e.,

$$p_r\ \left(\rho\right) = \sum_{i=1}^{N} P\left[m_i\right]\ \ p_r\ \left(\rho | s = s_i\right), \tag{2-5}$$

the optimum receiver chooses m_k to maximize the quantity

$$P\left[m_k\right] \quad p_r\left(\rho \mid m_k\right)$$

If the source distribution $P\left[m_i\right]$ is not known, the receiver may just maximize $p_r\left(\rho \mid m_k\right)$. This is known as a maximum likelihood receiver and is optimum if the message probabilities, $P\left[m_i\right]$, are equal.

These definitions of maximum a posteriori and maximum liklihood receivers apply equally well to vector channels and waveform channels whenever the channel output probability density conditioned on the input is known.

With this background, a simple code can be devised for a digital channel. Consider the system shown in Figure 2-2. The message is drawn from the binary set [0,1]. The coder transforms the binary message into another binary signal. The channel is binary symmetric which means that input and output alphabets are discrete and binary and that the probability of either a zero being changed into a one or vice versa is the same and is denoted by p. A binary symmetric channel (BSC) is shown in Figure 2-3. The decoder examines the corrupted signal and guesses whether to set $\hat{m}$ equal to zero or to one.

How should the coder transform the message, and how should the decoder guess the value of $\hat{m}$? If no coding is employed, an error in a single location can change a message bit and there is no way for the receiver to know an error has occurred (assuming the message bits are statistically independent). A logical first approach to coding is thus to let each message bit influence more than one coded, or channel, bit. The distribution of influence of each message bit in this manner is the basis of all error-correcting and error-detecting codes.

For purposes of example, the system in Figure 2-2 employs a repetition code, that is, each message bit is simply repeated n times. The message bits are assumed equi-probable and the decoder is, therefore, designed to choose for $\hat{m}$ the value m_k for which

$$P_r\left[\rho_n \mid m_k\right]$$

is a maximum. ρ_n is the received binary n-tuple. Assume that p, the BSC error probability is less than one half. (The

channel output symbols can always be labeled so that $p \leq 1/2$.) If ρ_n contains j ones and n-j zeros, then

$$P_r\left[\rho_n \mid 0\right] = p^j (1-p)^{(n-j)} \tag{2-6}$$

and

$$P_r\left[\rho_n \mid 1\right] = (1-p)^j \, p^{(n-j)} \tag{2-7}$$

The decoder computes a decision function, γ, where

$$\gamma = \frac{P_r\left[\rho_n \mid 0\right]}{P_r\left[\rho_n \mid 1\right]}$$

$$= \frac{p^j \, (1-p)^{(n-j)}}{(1-p)^j \, p^{(n-j)}}$$

$$= \left(\frac{p}{1-p}\right)^{2j-n} \tag{2-8}$$

$\hat{m}$ is set equal to zero if γ is greater than one and vice versa. Thus, the optimum decoder uses majority rule. To see this, let more than half of ρ_n be ones, i.e. $j > \frac{n}{2}$. Then γ is less than one, and $\hat{m} = 1$. If $\gamma = 1$, the decoder cannot correct the errors except by making a random choice which in itself has a probability of error equal to one half. To eliminate this possibility, the number of repetitions, n, is constrained to be odd.

The probability, P_E, of a message bit being erroneously estimated by the decoder is just the probability of channel erros in more than half the bits in ρ_n. That is,

$$P_E = \sum_{i = \left[\frac{n}{2} + 1\right]}^{n} \binom{n}{i} p^i \, (1-p)^{(n-i)} \tag{2-9}$$

where $\left[\frac{n}{2} + 1\right]$ is the greatest integer in $\left(\frac{n}{2} + 1\right)$.

The rate, R, of a code is the number of message bits per coder output bit. The rate of the simple repetition code discussed above is just

$$R = \frac{1}{n}, \quad \frac{\text{message bits}}{\text{coded bit}}$$

Figure 2-4 is a graph of P_E versus R for this code.[16] As would be expected, the probability of error decreases as the rate decreases (and n increases). Therefore, if the channel admits coded bits at a fixed rate, the repetition code can achieve vanishingly small error rates only at the cost of a vanishingly small information rate.

Fortunately, not all codes suffer this defect. In fact, Shannon[2] has shown that codes exist for which the probability of error is arbitrarily small for any rate less than C, the channel capacity. The existence of these codes is indicated by the lower curve in Figure 2-4. Shannon's result is non-constructive, i.e., it is proven by averaging over the results of all possible codes. For any of the practical codes which allow error rate to be reduced indefinitely, one or more of the following characteristics must be correspondingly increased indefinitely: decoder complexity, channel bandwidth, receiver storage, and decoding delay. Thus, although the ideal code has not been found, presently available codes are useful in reducing the error rate to a small, but non-zero, value.

Code Classes

For the purposes of this memorandum, codes are classified according to the manner in which the influence of each message bit is distributed among the channel signals and according to how the channel signals are detected by the receiver. The two code classes are block and convolutional. Optimal and sub-optimal detection of block codes are defined.

A code belongs to the block code class if the message symbol sequence and the channel symbol sequence can be segmented into disjoint intervals, or blocks, bearing a constrained relationship to each other. The constraint on this relationship is illustrated in Figure 2-5. Let g be a particular message block, and G a particular channel block. The other message blocks and channel blocks are represented by h and H, respectively. Each message block, g, must be related to a channel block, G, so that each symbol in g influences the contents of G but not H and each symbol in G is influenced by the contents of g but not h.

The constraints on a convolutional code are similar to those on block codes except that the message segments are not disjoint, but overlap, as shown in Figure 2-6. The contents of G_1 are influenced by only g_1 and g_1 influences only G_1.* Similarly for G_2 and g_2, G_3 and g_3, etc. The g_i form a sliding aperture, or window, along the message symbol sequence.

Assume the channel block, G in Figure 2-5, of a block code is received as G' after corruption by additive white Gaussian noise. If the detector correlates the entire G' with all possible forms of G, the detection is optimal. If the detector correlates subintervals of G' with the possible subintervals of G and coarsely quantizes the results of each correlation (usually into one bit), the detection is suboptimal.

The reason for choosing suboptimal detection over optimal detection in some cases will become clear in Section III.

If both the message symbols and the channel symbols are binary, common terminology exists to express the complexity and rate of either block or convolutional codes.

In the block code case, if the number of binary digits in g is k and the number in G is n, then the code is said to be an (n,k) code and to have rate (number of message bits per channel bit) of $\frac{k}{n}$.

In the convolutional code case, if G_i contains v bits, and the sliding window moves w bits per shift, the rate is

$$R = \frac{w}{v} = \frac{\text{message bits}}{\text{channel bit}}$$

Let L be the total number of sliding windows, or g_i, in which a particular message bit can appear. Then the constraint length, K, is defined as

$$K = Lv$$

The constraint length is just the total number of channel bits which can be influenced by one message bit.

The constraint length of a convolutional code and the block length, n, of a block code can be used with the reliability function (introduced in Section III) to bound the probability of error.

---

*This is true during the encoding of g_1.

III. THEORETICAL BOUNDS ON CODING PERFORMANCE

Shannon's Noisy Coding Theorem[2] states that for rates less than channel capacity, C, codes exist for which the error probability is arbitrarily small. The channel capacity is a function only of the channel and the constraints imposed by the modulation and demodulation equipment. These constraints usually include bandwidth and quantization of transmitted and receiver signals. This section discusses the capacities of some important channel models and the relationship between code complexity and probability of error.

Capacity

Shannon has proved that the capacity of a power constrained, approximately bandwidth constrained, continuous channel corrupted by additive white Gaussian noise is

$$C = W \log_2 \left[1 + \frac{P_s}{WN_o}\right] \quad \text{bits/sec} \qquad (3\text{-}1)$$

Reference 3 discusses the problem of defining bandwidth for this theorem; it is assumed, however, that the signal energy is concentrated in the band [-W, W]. The average signal power is P_s and the two sided power spectral density of the Gaussian process is $\frac{N_o}{2}$.

Obviously, the capacity increases indefinitely with increasing signal power. This is reasonable, since with increasing signal-to-noise ratio the receiver should be able to distinguish between more and more numerous signaling levels. The same relationship does not hold with bandwidth, as can be seen in the plot of capacity as a function of bandwidth in Figure 3-1. Even for an infinite bandwidth, the capacity does not exceed

$$C_\infty = \frac{P_s}{N_o} \log_2 e$$

$$= 1.44 \frac{P_s}{N_o} \qquad (3\text{-}2)$$

Therefore, by Shannon's Theorem, the message bit rate, R, in bits per second must satisfy,

$$R < C_\infty = 1.44 \frac{P_s}{N_o} \qquad (3\text{-}3)$$

Let E_b be the received energy per message bit and let T be the message bit duration. Then the rate, R, in bits per second is

$$R = \frac{1}{T} \qquad (3\text{-}4)$$

and

$$\frac{E_b}{N_o} = \frac{P_s T}{N_o} = \frac{P_s}{N_o} \frac{1}{R}$$

$$> \frac{P_s}{N_o} \frac{1}{1.44} \frac{N_o}{P_s}, \text{ by (3-3)}$$

$$= \frac{1}{1.44}$$

or

$$\frac{E_b}{N_o} > -1.6 \text{ dB.} \qquad (3\text{-}5)$$

This bound for the infinite bandwidth Gaussian Channel is known as Shannon's limit and is used as a reference for judging coding systems. A code requiring very little power in excess of Shannon's limit to achieve a low probability of error is a good code. Of course, no code operating with

$$\frac{E_b}{N_o} < -1.6 \text{ db}$$

can achieve a vanishing small probability of error, regardless of its complexity. To give a specific example, consider an uncoded coherent PSK system with a bit error rate requirement of 10^{-4}. The required $\frac{E_b}{N_o}$ is 8.4 dB. This is 10 dB greater than

Shannon's limit and, therefore, this uncoded system requires ten times the power required by an ideal coded system.

Shannon's Theorem applies to transmitters which are constrained only in average power and in bandwidth and to receivers in which the output of the detector can have a continuous range of values. Frequently, due to practical considerations, the transmitter is also constrained to transmit waveforms chosen from a finite set and the receiver detector output is quantized before the message estimate is made. This quantization of transmitter output and detector output tends to decrease the channel capacity, and the coarser the quantization, the greater the decrease in capacity. For example, if $\frac{E_b}{N_o}$ is +1.8 dB and the channel (including detector) has a continuous input and a continuous output, the capacity is 2 bits per second per unit bandwidth. If the channel input is constrained to be one of two possible waveforms (binary signaling) the capacity drops to 1.6 bits per second per unit bandwidth. If, in addition, the detector output is quantized into two levels (one bit quantization) the capacity drops further to 1.0 bit per second per unit bandwidth.[5]

A Gaussian channel with discrete inputs and outputs is a discrete memoryless channel, or DMC, which can be completely characterized by its probability transition matrix, P_{ij}. Here, P_{ij} is the probability that the channel output is i given that the input is j. Given the values of P_{ij}, the DMC capacity can be calculated directly by methods discussed in Reference 6. The result for the binary symmetric channel is

$$C_{BSC} = 1 + p \log_2 p + (1-p) \log_2 (1-p) \quad \text{bits/symbol} \quad (3\text{-}6)$$

The dimensions for this expression for capacity are bits per symbol, i.e., bits per BSC usage. If the BSC is used B times per second, the capacity is

$$B \cdot C_{BSC} \quad \text{bits/second}$$

In spite of capacity degradation, bandwidth constraints and quantization are usually accepted to facilitate implementation.

Reliability Functions

As stated in Section II, the probability of error of a code depends on its complexity. More specifically, upper and lower bounds on probability of decoding error decrease expcnentially with block length or with constraint length.

The probability that a block of a block code will be erroneously decoded, $P_B[e]$, is bounded by,[6]

$$A_1 e^{-n \tilde{E}_B(R)} \leq P_B[e] \leq A_2 e^{-n E_B(R)} \tag{3-7}$$

In Equation (3-7), n is the block length, the A_i are constants or slowly varying functions of n, and $\tilde{E}_B(R)$ and $E_B(R)$ are functions of the rate, R, called reliability functions.

The probability that one of the channel blocks, or G_i, will be erroneously decoded in a convolutional code, $P_c[e]$, is bounded by,

$$A_3 e^{-K \tilde{E}_c(R)} \leq P_c[e] \leq A_4 e^{-K E_c(R)} \tag{3-8}$$

Here, the A_i are slowly varying functions of K, the constraint length, and $\tilde{E}_c(R)$ and $E_c(R)$ are the reliability functions.

The lower bound in each of the above cases is a basic limit on the ability of any code to reduce the probability of error. The upper bound, on the other hand, is the probability of error randomly averaged over all possible codes. Therefore, there is at least one code which will perform as well as the upper bound.

A sketch of the reliability functions is shown in Figure 3-2. The curves shown are typical for a discrete memoryless channel. The reliability functions for an optimally detected block code are similar to those for a DMC and are not shown. In Figure 3-2, C is the channel capacity and

$$E_o(1, Q) = -\ln \sum_{j=o}^{J-1} \left[\sum_{k=0}^{K-1} P_k \sqrt{P_{jk}} \right]^2 \tag{3-9}$$

where

K = input alphabet size of the DMC

J = output alphabet size

P_k = input probability distribution

P_{jk} = probability that DMC output is j given that input is k.

Note that the reliability functions all go to zero at channel capacity and that the functions for convolutional codes are above those for block codes for all but very small rates.

The derivations of these curves are given in Reference 6. The upper and lower probability of error bounds for convolutional codes were derived by Yudkin[7] and Viterbi[8], respectively.

IV. CODING TECHNIQUES AND PERFORMANCE

This section introduces some specific coding techniques. These coding techniques are grouped into the classifications of optimally detected block, suboptimally detected block, and convolutional. Decoding methods are discussed and the performance of each code as determined by analysis or as demonstrated by computer simulation is given.

Waveform Coding

For the purposes of this memorandum, waveform coding refers to any optimally detected block coding. Since the transmitted messages are assumed equally probable, the optimum detector assigns to each received waveform the message waveform block with which it is most highly correlated.

Let $\left[s_i(t)\right]$, $o \leq t \leq T$, $i = 1, \cdots, M$, be a set of M equal energy block waveforms which are transmitted with equal probabilities. Let r(t) be the received waveform, that is, the transmitted waveform plus Gaussian noise. Then the optimum detector has the structure shown in Figure 4-1. The filters are matched to the message waveforms $\left[s_i(t)\right]$ and are sampled at the end of the message waveform interval, when $t = T$. The message estimate, $\hat{m}$, is just the message corresponding to the filter with the largest output.

This detector structure may be implemented more easily if the $\left[s_i(t)\right]$ are first represented in terms of orthonormal components. Let $\left[\phi_j(t)\right]$, $o \leq t \leq T$, $j = 1, \cdots, N$, be a set of orthonormal waveforms from which the message waveforms can be generated, i.e.,

$$\int_0^T \phi_i(t)\,\phi_j(t)\,dt = 1,\ i = j \qquad (4\text{-}1)$$
$$= 0,\ i \neq j$$

and

$$s_i(t) = \sum_{j=1}^{N} s_{ij}\,\phi_j(t),\ i = 1, \cdots, M. \qquad (4\text{-}2)$$

As shown in Reference 3, the optimum receiver chooses for the message estimate the message corresponding to the signal $s_i(t)$ which maximizes

$$\underline{r} \cdot \underline{s_i} = \sum_{j=1}^{N} r_j\,s_{ij} \qquad (4\text{-}3)$$

where

$$\underline{r} = \left[r_1\ r_2\ \cdots\ r_N\right]$$

$$\underline{s}_i = \left[s_{i1}\ s_{i2}\ \cdots\ s_{iN}\right]$$

and

$$r_j = \int_0^T r(t)\,\phi_j(t)\,dt \qquad (4\text{-}4)$$

This structure can be realized as shown in Figure 4-2. At first glance, this appears to be more complicated than the realization in Figure 4-1. However, it can be shown using the Gram-Schmidt orthogonalization procedure that $N \leq M$. If M is large and $N << M$, the second realization offers appreciable savings in matched filters.

Further savings in equipment result from proper choice of the orthonormal waveforms, $\left[\phi_j(t)\right]$. Consider the set of time disjoint pulse waveforms shown in Figure 4-3. Assuming that the pulses from different waveforms do not overlap and that the integral of $h^2(t)$ is unity, the waveforms are orthonormal. The optimum receiver may, therefore, be realized as shown in Figure 4-4. The filter is matched to the elementary pulse, h(t), and is sampled at the end of each elementary pulse interval. The values of the samples are just $r_1, r_2, r_3, \cdots, r_N$,

which can then be combined with the s_{ij} in the decision circuits to find the value of i which maximizes

$$\sum_{j=1}^{N} r_j \, s_{ij} \, .$$

It has now been shown that the problem of coding for the Gaussian channel reduces to the problem of coding for a vector channel, i.e. a channel which changes $\underline{s}_i$ into $\underline{r}$. This problem will be approached by displaying choices for the code vectors, s_i, and showing the effect of each choice on probability of error and bandwidth occupancy.

A choice of a set, s_i, determines a code. Three optimally detected blocks codes are considered: the orthogonal, biorthogonal, and transorthogonal. These codes are easily visualized in signal space as shown in Figure 4-5. In this figure, the coordinates along the ϕ_1 and ϕ_2 axes are just s_{i1} and s_{i2}. Orthogonal code points lie on the positive orthonormal axes, i.e., for the orthogonal code shown,

$$\underline{s_1} = \left[\sqrt{E_s}, \quad o\right] \tag{4-5}$$

$$s_1(t) = \sqrt{E_s} \quad \phi_1(t) + o \cdot \phi_2(t) \tag{4-6}$$

and similarly for $\underline{s_2}$. The signal energy is E_s for each code waveform, i.e.,

$$\int_o^T s_1^{\,2}(t) \, dt = \int_o^T s_2^{\,2}(t) \, dt = E_s \tag{4-7}$$

For any orthogonal code, N=M. The correlation of the code waveforms is

$$\int_o^T s_i(t) \, s_j(t) \, dt = E_s \quad , \quad i = j \tag{4-8}$$
$$= o \quad , \quad i \neq j$$

The biorthogonal code is just an orthogonal code plus its negative. Therefore, M = 2N and

$$\int_0^T s_i(t)\, s_j(t)\, dt = E_s \quad i = j \qquad (4\text{-}9)$$

$$= 0 \quad i \neq j,\ i \neq \bar{j}$$

$$= -E_s \quad i = \bar{j}$$

Here $\bar{j}$ is the index of the one waveform which is the negative of the waveform having index i.

A transorthogonal code is obtained by rotating and translating, in signal space, an orthogonal code in such a manner as to minimize the average energy required. For example, the transorthogonal code shown in Figure 4-5 is obtained from a N=M=3 orthogonal code. This does not change the probability of error and the average energy required is reduced by a factor of $\left(1-\frac{1}{M}\right)$. The transorthogonal (or simplex) code is believed to be optimum[10], although this has been proven only for $M \leq 3$. For any transorthogonal code

$$M = N + 1,$$

$$\int_0^T s_i(t)\, s_j(t) = E_s \quad , \quad i = j \qquad (4\text{-}10)$$

$$= -\frac{E_s}{M-1} \quad , \quad i \neq j$$

One method of constructing waveform codes employs Hadamard matrices. A Hadamard matrix is a square matrix whose elements are plus and minus ones and whose row vectors are mutually orthogonal. A Hadamard matrix of order two is

$$H_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Hadamard matrices of order 2^k, k = 2, 3, 4, ···, can be contructed from H_1 by the recursion relation

$$H_k = H_1 \times H_{k-1} \tag{4-11}$$

The operation on the right side of Eq. (4-11) is the Kronecker product of H_1 and H_{k-1}, i.e., each one in H_{k-1} is replaced by H_1 and each minus one is replaced by minus H_1. For example,

$$H_2 = H_1 \times H_1$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{4-12}$$

An orthogonal code can be constructed by choosing for the s_i the row vectors of a Hadamard matrix of order $2^k = M$ which has been multiplied by a scale factor (so that the signal energy is E_s). The corresponding biorthogonal code results from adding to the orthogonal code its complement. Finally, a transorthogonal code can be constructed by choosing for the s_i the row vectors of a Hadamard matrix of order $2^k = M$ which has been multiplied by a scale factor and from which the first column has been deleted. Matrix H_2 in (4-12) yields the codes shown in Figure 4-6.

The codes in Figure 4-6 do not seem to agree with the signal space definitions given earlier, e.g., the orthogonal code points, S_i, do not lie on the coordinate axes. However, this discrepancy is nothing more than a rotation of axes which affects neither the probability of error nor the correlation properties of the codes.

The waveform codes discussed above can be implemented with feedback shift registers as discussed in Reference 9.

An important constraint on the use of waveform coding is its bandwidth requirement. Rate was earlier defined as the ratio of message bits to coded bits. This definition can be modified to include the time required to transmit one code word, T. The rate in message bits per second is

$$R = \frac{\log_2 M}{T} \quad \text{bits/sec.}$$

The minimum carrier frequency separation of waveform code words which does not lead to interference can be shown to be[9]

$$\Delta F_{Min} = \frac{N}{2T}$$

This can be considered an effective bandwidth, W, i.e.,

$$W = \frac{N}{2T}$$

The bandwidth required per unit rate is

$$\frac{W}{R} = \frac{N}{2 \log_2 M}$$

For orthogonal codes, N = M, and

$$\frac{W}{R} = \frac{M}{2 \log_2 M}$$

For biorthogonal codes, M = 2N, and

$$\frac{W}{R} = \frac{M/2}{2 \log_2 M}$$

For transorthogonal codes, M = N + 1, and

$$\frac{W}{R} = \frac{M - 1}{2 \log_2 M}$$

Note that

$$\lim_{N \to \infty} \frac{W}{R} = \lim_{M \to \infty} \frac{W}{R} = \infty$$

for orthogonal, biorthogonal, or transorthogonal codes. Therefore, with a fixed information rate, any decrease in error probability requires an increase in bandwidth for these codes. This bandwidth expansion and the complexity of the weighting matrix in Figure 4-2 are the main drawbacks of these codes. Of course, by Shannon's Theorem, waveform codes exist for which the probability of error can be reduced to an arbitrarily small value with no increase in power or bandwidth. However, there is no known method for deterministically constructing these or easily decoding them.

Note, also, that biorthogonal codes require less bandwidth per unit information rate than either orthogonal or transorthogonal codes for M > 2.

For M = 32, the probability of word error is shown* in Figure 4-7 for orthogonal, biorthogonal, transorthogonal, and uncoded transmission[9] on an additive Gaussian noise channel having a uniform power spectral density of $\frac{N_o}{2}$. E_b is the energy per information bit. The probability of word error for uncoded transmission is the probability that the transmission of a block of five bits using antipodal signals, i.e., $s_1(t) = -s_2(t)$, results in one or more bit errors.

Ordinarily, the measure of error which is of concern is the probability of information bit error. This is shown[9] for biorthogonal codes of various lengths in Figures 4-8 and 4-11.

BCH Codes

BCH codes are suboptimally detected block codes. They were discovered by Bose and Chaudhuri[11] and independently by Hocquenghem.[12] The BCH decoder operates on symbols from a finite alphabet, usually binary. Thus the detector output for each received symbol must be coarsely quantized resulting in suboptimality. Referring to Figure 1-1, a BCH code system would not include the shift register between the encoder and the waveform generator. For example, if the BCH code is binary, the waveform generator converts each code symbol directly into one of

*In Figures 4-7, 4-8 and 4-11, $k-\log_2 M$.

two waveforms, s_o (t) or s_1 (t). A block diagram for the detector in a binary BCH code system is shown in Figure 4-9.

In general, the alphabet for a BCH code can contain any number of symbols. The binary BCH code is the most easily instrumented, however, and only the binary case will be considered in this section.

The coder in Figure 1-1 segments the information sequence into blocks. From each block parity check bits are calculated. A parity check bit is the modulo two sum of a particular subset of the information bits. The information bits and parity check bits constitute the code word. As discussed earlier, an (n, k) code has code words of length n, containing k information bits and n-k parity check bits.

The coder for a (31, 21) BCH code is shown in Figure 4-10. It consists of a shift register with feedback connections. Initially, the register is filled with zeros, switch A is closed, and switch B is open. Twenty-one information bits leave the coder as coded output and simultaneously enter the feedback shift register. The register is shifted in synchronism with the information bits. The information source is then halted, switch A is opened, and switch B is closed. The contents of the shift register, the parity check bits, are shifted out. This leaves the shift register in the all zero state and the process can repeat.

It is the selection of the feedback connections which determine the properties of the code and the amount of computation required to decode received messages. The procedure for choosing these connections can be developed in a mathematically elegant way from the concepts of modern algebra. It is the algebraic structuring of the code which allows practical decoding. Without this structuring, the decoder would require a codebook containing 2^n entries.

The most typical form of BCH codes can be defined as follows. Let $X_n, X_{n-1}, \cdots, X_1$ be the symbols in an (n, k) code word. X_n is the first symbol transmitted and X_1 the last. Then the sequence $X_n, \cdots, X_1$ is a BCH code word if, and only if[6],

$$\sum_{i=1}^{n} X_i \left[\alpha(s)\right]^{j(i-1)} = A(s)\ B(s)\ ,\ j = 1, \cdots, d-1 \qquad (4\text{-}13)$$

$$2 \leq d \leq n$$

In (4-13), $\alpha(s)$ is a polynomial of degree less than m, where $m = \log_2 (n + 1)$, B(s) is an irreducible polynomial of degree m, and A(s) can be a polynomial of any degree. All polynomials have binary coefficients and all multiplications and additions follow the rules of binary arithmetic. The parameter d is related to the error correcting ability of the code.

It can be shown[13] that for any m and t there is a binary BCH code of length $n = 2^m-1$ which corrects all combinations of t or fewer errors within a transmitted block and has no more than mt parity check bits.

The performance of a (73,45) BCH code is shown[18] in Figure 4-11. The channel is Gaussian and the detector output is binary. The decoder is not a maximum likelihood decoder, but a threshold decoder. The threshold decoder has nearly as good performance as the maximum likelihood decoder with less complexity. The concept of threshold decoding is discussed in this report in connection with convolutional codes.

Convolutional Codes

As discussed in Section II, there is a basic difference between block and convolutional codes in the way the influence of a message bit is distributed. An implementation of a simple convolutional encoder is shown in Figure 4-12.

In operation, message symbols are shifted into the shift register one at a time. Each shift causes the contents of the register to shift one position to the right, the right most symbol being discarded. After each shift, the channel switch samples the outputs of the adders.

The theory of convolutional codes is valid for non-binary as well as binary alphabets. However, because of implementational considerations, binary alphabets are ordinarily used, in which case the adders perform addition modulo two.

In Figure 4-12, two channel symbols are produced for each message symbol and the rate is one half. Any rational rate is possible with a convolutional coder. Assume there are j shift registers each of which is connected with v adders. During each cycle, j information symbols are accepted by the coder, one into each shift register. The sum of the outputs of the first adders for each register is read out. The sum of the second adders is read out, and so on, producing v output symbols per cycle. The rate is then

R = message symbols per output symbol

= j/v

Most analysis and simulations in the literature concern rate one half and rate one third codes.

Constraint length has been defined as the number of channel symbols over which one message symbol has influence. For the generalized encoder in the last paragraph, the constraint length, K, is just

$$K = v\ L$$

where L is the length of each shift register. For example, the constraint length of the encoder in Figure 4-12 is six.

A helpful visual aid in understanding the operation of the decoders to be discussed is the tree structure of a convolutional code. The tree structure generated by the encoder of Figure 4-12 is shown in Figure 4-13. Encoding a sequence of message symbols corresponds to choosing a path through the tree structure. Each message bit corresponds to moving from left to right one branch length. If the message bit is zero,the upward branch is chosen and vice versa. For example, in Figure 4-14 are shown the states of the shift register and the output symbols for the message sequence 0101. This corresponds to the path indicated by the dotted line in Figure 4-13. The shift register is assumed initially to contain all zeros. The tree structure can, of course, be extended indefinitely.

At the end of a finite length message sequence, special steps must be taken to insure that the protection of the last constraint length of message bits (the last jL message bits) is equal to that of earlier bits. This is usually accomplished by suffixing the message sequence with L - 1 zeros. The last L - 1 nodes of the code tree then have only single branches leaving them, and the shift register is ready for a new message sequence to begin.

In certain of the decoders to be discussed, it is desirable to periodically terminate the coding process with L - 1 zeros whether or not the message has ended and immediately restart. This resynchronizes the decoder if a catastrophic decoding failure (such as buffer memory overflow) has occurred.

All of the decoders of convolutional codes discussed in this section are known as probabilistic decoders with the exception of the threshold decoder. A probabilistic decoder contains a replica of the encoder and decoding consists of assuming a tentative message, encoding it, and comparing the result with the actual received sequence. If the two coded sequences are similar enough, the tentative message is assumed

correct; if not, another tentative message is tried. The basis for judging similarity is the tree path value or path metric. The exact form of the metric varies from algorithm to algorithm, but it is generally a measure of likelihood of a particular path through the tree given the received sequence.

Sequential Decoding

Both sequential decoding algorithms discussed in this report are limited to the same range of code rates and can attain arbitrarily low probabilities of undetected error.

Each technique requires memory storage which, in periods of high noise intensity, can fill and overflow. The effects of memory overflow will be discussed separately for each algorithm. It can be shown that the average number of computations per message bit tends to infinity with increasing message sequence length for rates above the computation rate, R_{COMP}, and is bounded above by a constant for rates less than R_{COMP}. The computation rate is a function only of channel transition probabilities and exceeds one half the capacity, or $\frac{C}{2}$, for all non-pathological memoryless channels. For example, Figure 4-15 shows R_{COMP}/C as a function of crossover probability, p, for a binary symmetric channel.

For either algorithm the average amount of computation required to decode one message bit is bounded above by a constant for any constraint length (assuming $R < R_{COMP}$). Therefore, the probability of undetected error can be made to approach zero by increasing the constraint length without incurring excessive computation.

The algorithms to be discussed are the Fano[14] and the Jelinek.[15] Fano's algorithm has been known since 1963 and has been thoroughly analyzed and simulated. Jelinek's algorithm was published in 1969 and very little work on it has appeared in the literature.

Sequential Decoding - Fano Algorithm

Given a received sequence, a sequential decoder considers the tree paths which might have been transmitted and selects the path which is most like the received sequence. More specifically, the path is chosen for which the path value, Γ, is

maximum. The path value is the sum of bit metrics, λ_t, of the bits on the path. The bit metric of the bit at time t, λ_t, is defined as

$$\lambda_t = \log_2 \frac{P_{ji}}{\sum_{k=1}^{2} P_k P_{jk}} - \beta$$

where

P_{ji} = probability of quantized detector output at time t being j given that i is transmitted,

P_k = probability that k is transmitted,

and

β = a constant called the bias.

Thus the bit metric is just the mutual information[16] between channel input and detector output less a constant. To better appreciate the meanining of the path value, assume the detector is one bit quantized (yielding a BSC) and that the P_k are equal. The path value reduces to a constant minus the Hamming distance. The Hamming distance between two binary sequences is just the number of bit locations in which the sequences differ.

As an example, consider the path indicated by the dotted line in Figure 4-13. Assume the detector employs hard decision and that the path value is

$$\Gamma = \sum_{t=1}^{\ell} \lambda_t = \ell\beta - h(\ell)$$

where

ℓ = length of path

and

$h(\ell)$ = Hamming distance between tree path and received sequence.

β = bias

If the detector output is

00 10 01 11

the path value of the dotted path is

$$\Gamma = 8\beta - 2$$

The mutual information metric has the property that if the transmitted sequence is received error free, the path value of the correct tree path is greater than the path value of any other tree path. This is true for any sequence length (tree penetration depth). If errors occur, the path value of the correct path may dip below the values of some incorrect paths. However, if the code constraint length is great enough, the path value of the correct path will eventually surpass all others. The function of the decoding algorithm is, thus, to select the path which ultimately has the greatest path value. Examples of path values for correct and incorrect paths are shown in Figure 4-16. Only a few of the incorrect paths are shown. Note that noise causes the correct path value to dip below another path value for a limited interval.

To understand the Fano algorithm for searching the code tree, assume the code tree is plotted graphically with paths running from left to right and with the vertical coordinate of each node equal to its path value, Γ, as in Figure 4-16. The algorithm uses an imaginary pointer which points to specific nodes in the tree. A running threshold is used to test the progress of the search. The running threshold is started at zero and is increased by multiples of some increment, Δ, each time the path value increases so that the threshold is always just below the node under consideration. A flow diagram of Fano's algorithm is shown in Figure 4-17.

The algorithm is divided into two sections: the forward mode and the back-up and search mode. If the path value of the correct path increases monotonically with a greater slope than any incorrect path, the algorithm remains in the forward mode, tightening the threshold after each step. If the slope of the correct path becomes negative or if it is less than that of an incorrect branch at some mode, the algorithm may start to follow an incorrect path. This will soon become apparent from the poor performance of the path value. The algorithm then enters the back-up and search mode to regain the correct path. An examination of Figure 4-17 should convince the reader that the algorithm will eventually select a path whose path value ultimately increases.

Obviously, the progress of the decoder through the code tree is highly variable. Since the data is transmitted at a constant rate, a buffer, such as that shown in Figure 4-18, must be used. The symbols from the detector output enter shift registers through the commutator. The first symbol in each branch enters the top shift register and the second enters the next register. The buffer in Figure 4-18 accommodates a rate 1/2 code; for rate $\frac{1}{v}$, v shift registers are needed for the input section of the buffer. Below the v input shift registers is the output shift register where the tentatively decoded bits are stored. The depth of search pointer indicates the input branch and tentatively decoded bit with which the decoder is concerned at any given time. As the input branches and tentative decisions flow from left to right, the depth of search pointer usually remains near the left end of the buffer. After a burst of noise, the pointer retreats to the right, erasing previous tentative decisions and searching for the correct path. If the depth of search pointer reaches the right end of the buffer, buffer overflow is said to have occurred.

When buffer overflow occurs, both the encoder and decoder must be resynchronized. This is done by truncating the transmission, inserting a known sequence into both the encoder and the decoder, and restarting the transmission. The known sequence may be inserted periodically to facilitate resynchronization; it may be inserted at the request of the decoder if the channel is two-way; it may never be inserted if the received symbols can be recorded and decoded off-line.

The probability of buffer overflow has been determined both analytically and by computer simulation. If the examination of one tree node by the decoder is defined as one computation, the probability that the number of computations required to decode D branches exceeds U, $P_D(U)$, can be shown to be[17]

$$P_D(U) = A(D,\alpha)\ U^{-\alpha},\ U>>1 \qquad (4\text{-}14)$$

In Equation (4-14), $A(D,\alpha)$ varies linearly with D over a large range and α is defined implicitly by

$$\frac{E_o(\alpha)}{\alpha} = \frac{1}{v} = R \qquad (4\text{-}15)$$

$E_o(\alpha)$ is Gallager's zero rate exponent.[6] For the binary input channel,

$$E_o(\alpha) = -\ln \sum_i \left[1/2 \sum_j P_{ij}^{1/(1+\alpha)}\right]^{1+\alpha} \qquad (4\text{-}16)$$

As usual, R is the code rate and P_{ij} is the DMC transition probability. A distribution such as that in Equation (4-14) is known as a Pareto distribution with α being the Pareto exponent.

Lumb[18] has determined $P_D(U)$ by computer simulation. Results for rate 1/2 codes of constraint lengths 30, 50 and 70 are shown in Figure 4-19. The number of bits decoded, D, was 224. The value chosen for $\frac{E_b}{N_o}$ was 2.66 dB and the detector output was three-bit quantized.

Note that the value of $P_D(U)$ increases with increasing constraint length (with U held constant). This is because a noise burst which may cause a very long search in a long constraint length code merely causes an undetected error in a short constraint length code.

Blustein and Jordan[19] have found an approximation to $P_1(U)$, the probability that the number of computations required to decode one bit exceeds U, from computer simulations.

$$P_1(U) \cong 3.16^{\left(\frac{R}{R_{COMP}} - 1\right)} U^{\left(-2.9 + 2\frac{R}{R_{COMP}}\right)} \qquad (4\text{-}17)$$

Equation (4-17) may be used to approximate the probability of buffer overflow, P_{BO}. Assume that the depth of search pointer is at the left end of the buffer in Figure 4-18. Denote by σ the number of computations the decoder can perform during the reception of one branch. If the decoder makes σB computations in the search mode and fails to advance one node deeper into the tree, the pointer will have reached the right end of the buffer and overflow will occur. The probability of this is $P_1(\sigma B)$. Because of the assumption that the pointer starts at the left end of the buffer, $P_1(\sigma B)$ is, strictly speaking, a lower bound to

P_{BO}. It can be shown by simulation[3] that when $P_1(\sigma B)$ is small that

$$P_{BO} \cong P_1(\sigma B) \tag{4-18}$$

A property of the Pareto distribution is that the expected value of a Pareto distributed random variable is finite only if the Pareto exponent is greater than one. In particular, the expected number of computations per decoded information bit is finite only if $\alpha>1$. This is equivalent to maintaining $\frac{E_b}{N_o}$ large enough so that

$$R < R_{COMP} \tag{4-19}$$

It can be shown[17] that in the case of infinite bandwidth and non-quantized detector output the value of $\frac{E_b}{N_o}$ required to maintain $R < R_{COMP}$ is 1.4 dB, or 3 dB above Shannon's limit. For finite bandwidth and quantized detector output $\frac{E_b}{N_o}$ must be even higher.

Since the computation required per information bit is bounded above as constraint length increases, the probability of undetected error can easily be reduced to a negligible value. Undetected errors are therefore not a problem with sequential decoding, and curves such as shown in Figure 4-11 are not shown.

Sequential Decoding - Jelinek Algorithm

Jelinek[15] has presented an alternative algorithm for searching a code tree. Just as with Fano's algorithm and any other sequential decoding algorithm, the code rate must be less than R_{COMP}; otherwide, the average computation per bit tends to infinity with increasing message length. Again, the probability of undetected error can be reduced to a negligible value by increasing the constraint length without a corresponding increase in computation.

The strategy employed by Jelinek's algorithm for searching the tree is distinctly different from that of Fano's algorithm. The message is assumed to be segmented into blocks N bits long. Let $P(\underline{s})$ be the probability that the initial portion of the message sequence was the binary sequence $\underline{s}$ given

all of the received block of N branches. The algorithm states:

1. Compute P(0) and P(1). Place these probabilities and the associated s sequences into an ordered memory, or stack, with the higher probability sequence at the top of the stack.

2. Remove the top or highest probability sequence s_o, from the stack. Extend s_o one bit (obtaining s_o0 and s_o1) and calculate the two new probabilities.

3. Place s_o0 and s_o1 and the associated probabilities in the stack in such positions to maintain monotonically decreasing probability order.

4. Repeat steps 2. and 3. until the top sequence is N bits long. This is the decoded block.

Initially, the memory stack is empty and each decoding step (after step 1.) adds one node to the contents of the stack. Since the stack is of finite size, and in any practical situation will have a capacity less than 2^n nodes, it must at some point overflow. On the next step after the stack has been filled, the bottom or least likely node is pushed from the bottom of the stack and thrown out. However, an index is maintained at the value of the highest probability of any deleted node. If this index ever exceeds the probability of the top node in the stack, a deleted node may have been on the correct path and an error is assumed to have been made. Just as in the case of Fano decoding, buffer overflow is catastrophic.

Jelinek has shown that for $R=R_{COMP}$, a stack memory with a capacity of 10N is sufficient to prevent catastrophic overflow in more than half the blocks. For smaller R, the buffer requirements are less.

In comparison with the Fano algorithm, the Jelinek algorithm requires less computation. For example[15], at $R=R_{COMP}$, Fano's algorithm requires an average of 13 computations per node compared to 2 for Jelinek's algorithm. The difference is smaller for smaller R, and is negligible at $R=0.8\ R_{COMP}$.

One explanation for the speed advantage of the Jelinek algorithm is that the decoder need never stop and wait on new

data. In a Fano decoder, if the pointer is at the left end of the buffer, no more computation can occur until the next branch of symbols has been received. In the Jelinek decoder, if s_o, the most probable sequence, is already as long as the amount of received signal, one of the shorter, less probable, sequences further down in the stack can be removed and extended. In this manner, the decoder is never idle.

Based on preliminary findings, the Jelinek algorithm appears to be a promising alternative to the Fano algorithm.

Viterbi Algorithm

Among all the decoders of convolutional codes discussed in this report, the Viterbi is the only optimum or maximum likelihood decoder. That is, its probability of error is the same as that of a decoder which calculates the path value of each of the 2^N paths in the tree and chooses the largest. This is accomplished with a complexity which is proportional only to 2^L (L is the length of the convolutional encoder shift register).

For the basic Viterbi algorithm, the message is segmented into blocks of length N. The $i^{\underline{th}}$ information symbol is denoted by A_i, the $i^{\underline{th}}$ received branch by $\underline{r}_i$. The decoder first computes the likelihood functions, $\prod_{i=1}^{L} P(\underline{r}_i | A_i)$, for all of the 2^L information sequences,

$$(A_1, A_2, \cdots, A_L)$$

For each of the 2^{L-1} sequences $A_2, \cdots, A_L$, the decoder chooses from the two sequences,

$$(0, A_2, \cdots, A_L) \qquad (4\text{-}20)$$

and

$$(1, A_2, \cdots, A_L), \qquad (4\text{-}21)$$

the sequence with higher likelihood. This sequence is called the survivor. For each possible combination of $A_2, \cdots, A_L$ there will be one survivor; therefore there are 2^{L-1} survivors, each of the form

$$(\alpha_1, A_2, A_3, \cdots, A_L). \qquad (4\text{-}22)$$

α_1 will be 0 or 1 depending on the following $A_2, \cdots, A_L$.

The decoder then uses the next received branch $\underline{r}_{L+1}$ to compute $P(\underline{r}_{L+1}|A_{L+1})$ for $A_{L+1} = 0$ and $A_{L+1} = 1$. With this information the decoder can choose from the two sequences

$$(\alpha_1, 0, A_3, A_4, \cdots, A_{L+1}) \tag{4-23}$$

$$(\alpha_1, 1, A_3, A_4, \cdots, A_{L+1}) \tag{4-24}$$

the one having the higher likelihood. This choice is made for each of the 2^{L-1} sequences $(A_3, \cdots, A_{L+1})$. The value of α_1 in (4-23) and (4-24) can be determined from the survivors in (4-22). The 2^{L-1} survivors now have the form

$$(\alpha_1, \alpha_2, A_3, A_4, \cdots, A_{L+1}) \tag{4-25}$$

This process continues, yielding a general form for the survivors of

$$(\alpha_1, \cdots, \alpha_{k-L+1}, A_{k-L+2}, \cdots, A_k) \tag{4-26}$$

for $k < N$. There is a survivor for each of the 2^{L-1} possible combinations

$$(A_{k-L+2}, \cdots, A_k) \tag{4-27}$$

After the N information bits are encoded, a tail sequence of L-1 known bits are encoded. At each decoding step for which $N<k \leq N+L-1$, the number of survivors is halved. After the decoding step with $k = N+L-1$, only one survivor remains, and its first N bits are declared the correct message. The decoder then begins on the next tree.

For implementation, a simplification can be made on the Viterbi algorithm which reduces memory requirements. When $k=N$, the total number of symbols in all the 2^{L-1} survivors is $N\,2^{L-1}$. Thus the tree length, N, is limited in the case of the ideal Viterbi algorithm by the size of the available memory. However, it has been found[20] that the initial portions of the survivors are usually identical to within five constraint lengths of k. Therefore, at each decoding step, the α_i decoded five constraint lengths earlier can be declared correct and removed from each survivor. This strategy also eliminates the necessity to terminate the code tree.

If the bit to be declared correct and removed from the survivors is not the same in each survivor, then the bit from only the survivor with highest likelihood can be used[21].

The storage requirement is $N\ 2^{L-1}$ for the ideal case or approximately $5\ L \cdot 2^{L-1}$ in the practical case. This dependence on the constraint length, L, limits the Viterbi decoder to use with short constraint length codes. For example, $5\ L\text{-}2^{L-1}$ for L = 15 is about 1.2 million.

An advantage of the Viterbi algorithm is that it decodes at a fixed rate with no possibility for catastrophic failure such as buffer overflow. It is therefore quite useful for real time application.

Heller[21,22] has simulated the performance of a Viterbi decoder with rate 1/3 codes and 3 bit detector quantization. The results for a code with a constraint length of 24 is shown in Figure 4-11. A code of constraint length K and rate R is denoted by (K, RK).

Ng Algorithm

Ng[23] has presented a decoding algorithm similar to the Viterbi algorithm, which he calls minimum Hamming distance (MHD) decoding. As in the Viterbi algorithm, the likelihood functions are first calculated for each of the 2^L sequences,

$$(A_1, A_2, \cdots, A_L).$$

Instead of forming survivors, the first symbol of the sequence with highest likelihood is declared correct and decoded. The likelihood functions are then calculated for all 2^L sequences

$$(A_2, A_3, \cdots, A_{L+1}).$$

Again, the first symbol, A_2, of the most likely sequence is declared correct and decoded. This procedure is repeated indefinitely with no need to terminate the tree.

Ng also discusses[23] some clever and involved ways for greatly reducing the computation required to calculate the likelihood functions of the sequences.

This algorithm is not optimal but it obviously requires less storage than the Viterbi algorithm.

Threshold Decoding

Threshold decoding does not depend on the tree structure of convolutional codes and can be used equally as well with block codes. In contrast to purely algebraic block decoders, threshold decoders can make use of the statistical information regarding the received bits which is available at the receiver.

Assume that the convolutional code is binary and systematic, i.e., the information bits appear unchanged among the parity bits of the coded output. Any parity bit, say p_a, can be written as the mod-2 sum of a set of information bits i_{a_j}

$$p_a = \sum_{j=1}^{n_a} i_{a_j} \qquad (4\text{-}28)$$

or

$$0 = \sum_{j=1}^{n_a} i_{a_j} + p_a \qquad (4\text{-}29)$$

in which all additions are mod-2. The a_j are the indices of the information bits whose sum is p_a. The sum of (4-29), when calculated from the received bits becomes,

$$\sum_{j=1}^{n_a} (i_{a_j} + e_{a_j}) + (p_a + e_a)$$

$$= \sum_{j=1}^{n_a} e_{a_j} + e_a \qquad (4\text{-}30)$$

in which e_i is the error (0 or 1) in the $i^{\underline{th}}$ bit. Notice that the i_{a_j} and p_a have dropped out because of Equation (4-29). Any mod-2 sum of received bits from which the information and parity bits drop out leaving only the error bits (as in Equation (4-30)) is called a parity check, A.

A set of parity checks $(A_1, A_2, \cdots\cdot, A_J)$ is said to be orthogonal on error bit e_a if e_a appears in each parity check of the set and no other error bit appears in more than one of the parity checks. In a threshold decoder, an error bit estimate, e_a^*, is formed using only the set of parity checks orthogonal on e_a.

Threshold decoding has two common forms, majority decoding and APP decoding[24]. In majority decoding, the decoding rule is

$$e_a^* = 1 \quad \text{if and only if} \quad \sum_{j=1}^{J} A_j > \frac{J}{2} \tag{4-31}$$

APP decoding (A Posteriori Probability decoding) makes use of the statistical information concerning the received bits. Its decoding rule is

$$e_a^* = 1 \quad \text{if and only if} \quad \sum_{j=1}^{J} w_j A_j > T \tag{4-32}$$

where

$$w_j = 2 \log \frac{P(A_j = e_a)}{P(A_j \neq e_a)}$$

and

$$T = 1/2 \ (w_1 + w_2 + \cdot\text{-} + w_J) + \log \frac{P(e_a = 0)}{P(e_a = 1)}$$

Although threshold decoding is sub-optimal, APP is superior to majority decoding due to its use of statistical information.

The performance of a rate 1/2 convolutional code[18] of constraint length 44 is shown in Figure 4-11.

V. CODING APPLICATIONS AND CONCLUSIONS

Operational Coding Systems

Four NASA applications of the codes discussed in this report are summarized:

1. The Mariner, 1969, Mars probes used six bit (M=64) biorthogonal waveform codes[26] at an information rate of 16.2 kilobits/sec. An information bit error probability of 5×10^{-3} was achieved with an $\frac{E_b}{N_o}$ of 3.00 dB.

2. NASCOM uses BCH codes to protect Apollo command data transmitted from the Manned Spacecraft Center (MSC) in Houston to remote sites via Goddard Space Flight Center (GSFC) in Greenbelt, Md. The basic coding unit between MSC and the remote sites is a (57, 30) BCH code. Between MSC and GSFC there is additional encoding into a (600, 567) BCH code. Error detection rather than error correction is used resulting in a probability of undetected error[27] of less than 4×10^{-13}.

3. The Pioneer D spacecraft uses a rate 1/2 convolutional code of constraint length 50 with sequential decoding. At the arbitrary comparison point of 10^{-4} information bit error probability, the convolutional code has a 5.1 dB gain over no coding.[18]

4. The Interplanetary Measuring Platform, or IMP I, will be launched in October, 1970 into a solar orbit to measure fields and solar particles.[28] Its telemetry system will use a rate 1/2 convolutional encoder of constraint length 96 with a Fano algorithm. In case of buffer overflow, a frame of 1024 information symbols is deleted. The system achieves an information bit error probability of 10^{-5} at an $\frac{E_b}{N_o}$ of 4 dB with a deletion rate of less than 1%.

Conclusions

The conclusions of this survey are:

1. The code reliability functions of Figure 3-2 imply that convolutional codes have error-correcting properties superior to those of block codes.

2. Prime candidates for non-real time decoding of convolutional codes are Jelinek and Fano sequential decoders. These sequential decoders are unsuitable for real time operation because of their highly variable decoding rate and possibility of catastrophic failure. Preliminary results indicate that the Jelinek decoder may require less computation than the Fano.

3. Prime candidates for real time decoding of convolutional codes are the Viterbi and the Ng decoders. Preliminary results indicate that the Ng decoder may achieve lower probability of error for a given amount of computation than the Viterbi decoder.[25] This is possible because even though the Ng decoder is suboptimal, it can decode, with the same amount of computation, a longer constraint length code than can the Viterbi.

Brian Tunstall

2034-BPT-mbr

B. P. Tunstall

Attachments

REFERENCES

1. Viterbi, A. J., Principles of Coherent Communication, McGraw-Hill, New York, 1966.

2. Shannon, C. E., "A Mathematical Theory of Communication", Bell System Technical Journal, Vol. 27, 1948, pp.379-423 (Part I), pp. 623-656 (Part II).

3. Wozencraft, J. M., and Jacobs, I. M., Principles of Communication Engineering, Wiley, New York, 1965.

4. Lucky, R. W., Salz, J., and Weldon, Jr., E. J., Principles of Data Communication, McGraw-Hill, New York, 1968.

5. Lefkowitz, Howard, Project Manager, Final Report for the Study of Sequential Decoding Techniques for Spacecraft Telemetry Systems, prepared by Communications and Systems, Inc., Falls Church, Va., for Goddard Space Flight Center, Greenbelt, Md., Contract No.: NAS5-11503, June, 1968.

6. Gallager, R. G., Information Theory and Reliable Communication, Wiley, New York, 1968.

7. Yudkin, H. L., Channel State Testing in Information Decoding, Sc.D. Thesis, Dept. of E. E., M. I. T., Cambridge, Mass., 1964.

8. Viterbi, A. J., "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", I.E.E.E. Trans. Inform. Theory, Vol. 13, No. 2, pp. 260-269, April, 1967.

9. Golomb, S. W., (Ed.), Digital Communications with Space Applications, Prentice-Hall, Englewood Cliffs, N. J., 1964.

10. Schalkwijk, J. P. M., "Recent Developments in Feedback Communication", Proc. I.E.E.E., Vol. 57, No. 7, pp.1242-1249, July, 1969.

11. Bose, R. C., and Ray-Chaudhuri, D. K., "On a Class of Error-Correcting Binary Group Codes", Inform. and Control, Vol. 3, pp. 68-79, 1960.

12. Hocquenghem, A., "Codes Correcteurs D'erreurs", Chiffres, Vol. 2, pp. 147-156, 1959.

References

13. Peterson, W. W., Error Correcting Codes, M.I.T. Press, Cambridge, Mass., and Wiley, New York, 1960.

14. Fano, R. M., "A Heuristic Discussion of Probabilistic Decoding", I.E.E.E. Trans. Inform. Theory, Vol. 9, pp. 64-74, 1963.

15. Jelinek, F., "Fast Sequential Decoding Algorithm Using a Stack", I.B.M. Journal of Research and Development, Vol. 13, pp. 675-685, Nov., 1969.

16. Abramson, N., Information Theory and Coding, McGraw-Hill, New York, 1963.

17. Jacobs, I. M., "Sequential Decoding for Efficient Communication from Deep Space", I.E.E.E. Trans. Communication Theory, Vol. 15, pp. 492-501, August, 1967.

18. Lumb, D. R., class notes from short course, "Recent Advances in Space Communications" given by University of California, Los Angeles, July 8-18, 1968.

19. Bluestein, G., and Jordan, K., "An Investigation of the Fano Sequential Decoding Algorithm by Computer Simulation", M.I.T., Lincoln Laboratory, Lexington, Mass., Group Report 62G-3, July, 1963.

20. Gates, H., Martin Marietta Co., Denver, Colorado, private communication, November, 1969.

21. Heller, J. A., "Sequential Decoding: Short Constraint Length Codes", Supporting Research and Advanced Development, Space Programs Summary 37-54, Vol. 3, pp. 171-177, Jet Propulsion Laboratory, Pasadena, California, Dec., 1968.

22. Heller, J. A., "Sequential Decoding: Improved Performance of Short Constraint Length Convolutional Codes", Supporting Research and Advanced Development, Space Programs Summary 37-56, Vol. 3, pp. 82-83, Jet Propulsion Laboratory, Pasadena, California, April, 1969.

23. Ng, W., and Pfeiffer, P. E., "Minimum-Hamming-Distance Decoding of Single-Generator Binary Convolutional Codes", Inform. and Control, Vol. 13, pp. 295-315, 1968.

24. Massey, J. L., Threshold Coding, M.I.T. Press, Cambridge, Mass., 1963.

References

25. Ng, W., Lockheed Electronics Co., Houston, Texas, private communication, October, 1969.

26. Posner, E. C., Combinational Structures in Planetary Reconnaissance, presented at the Symposium on Error-Correcting Coder at the Mathematics Research Center, U. S. Army, University of Wisconsin, Madison, Wisconsin, May 6-8, 1968.

27. Tunstall, B. P., Performance of Error Detecting Coding on NASCOM Data Circuits, TM68-2034-7, June 7, 1968.

28. Sos, John, Goddard Space Flight Center, Greenbelt, Md., private communication, November, 1969.

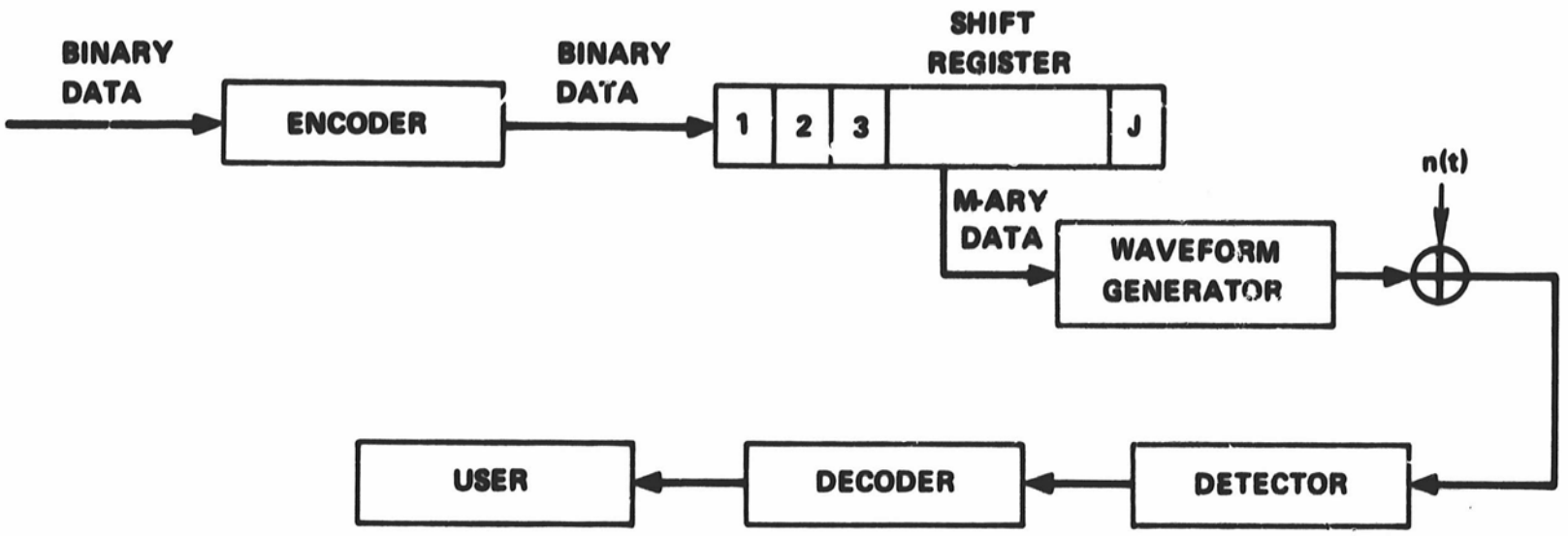


FIG. 1-1 SYSTEM BLOCK DIAGRAM

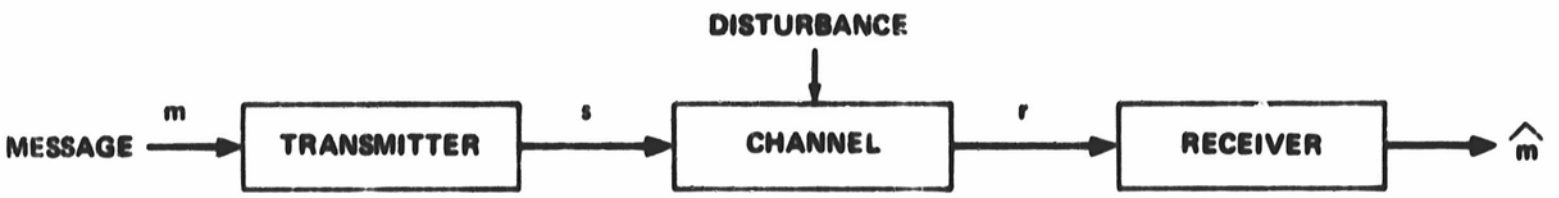


FIG. 2-1 SIMPLIFIED COMMUNICATION LINK

FIG. 2-2 SIMPLE CODED SYSTEM

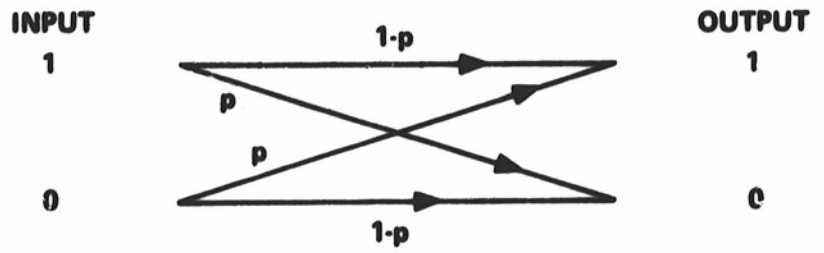


FIG. 2-3 TRANSITION PROBABILITIES OF A BINARY SYMMETRIC CHANNEL

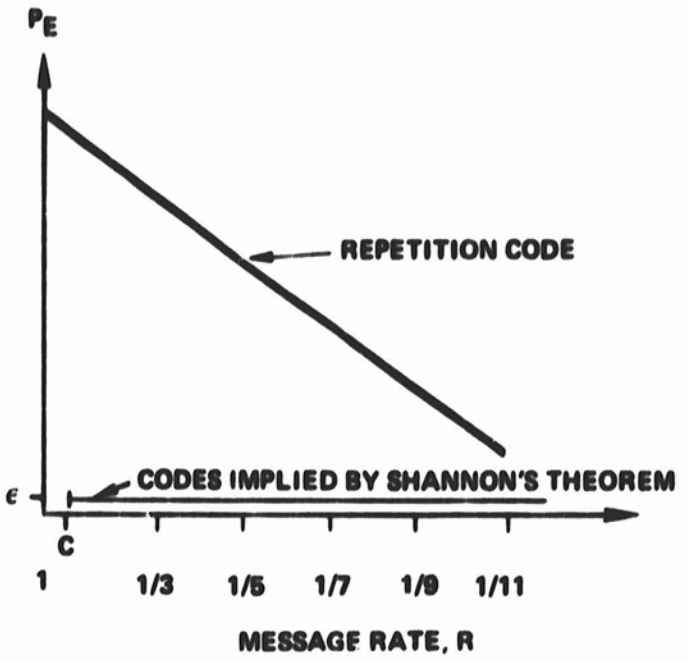


FIG. 2-4 PERFORMANCE OF REPETITION CODE

FIG. 2-5 BLOCK CODE CLASS

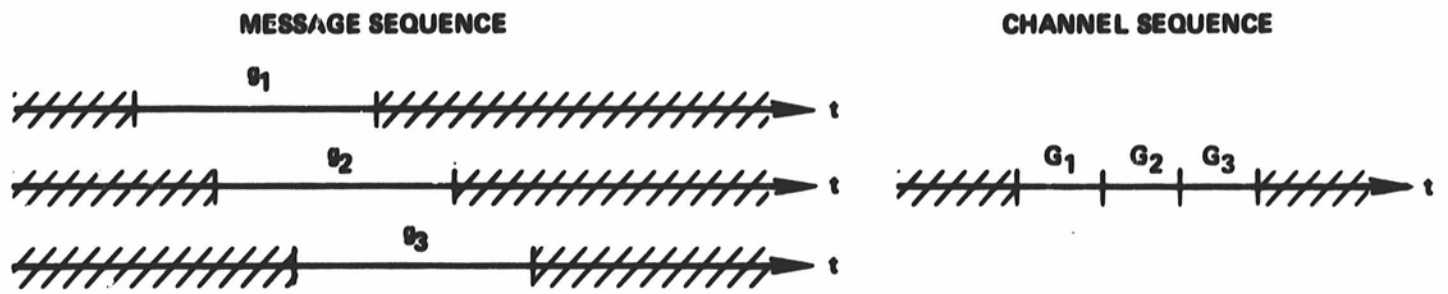


FIG. 2-6 CONVOLUTIONAL CODE CLASS

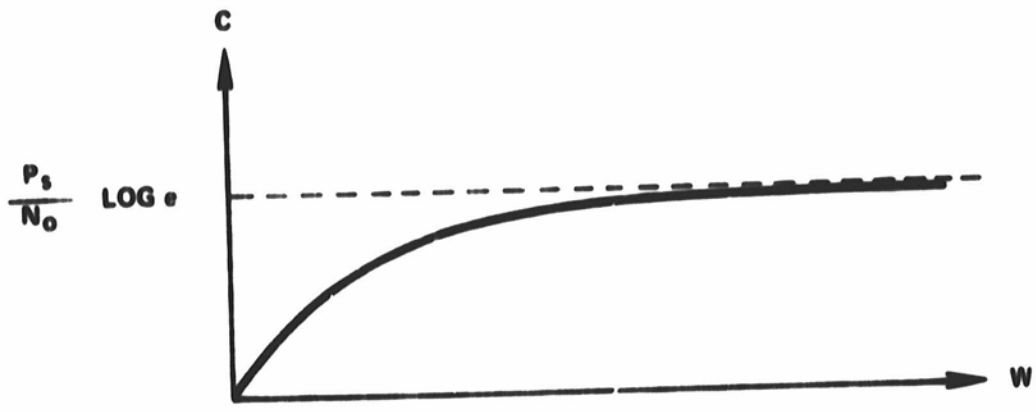


FIG. 3-1 CAPACITY OF WHITE GAUSSIAN CHANNEL WITH BANDWIDTH W

RELIABILITY FUNCTION

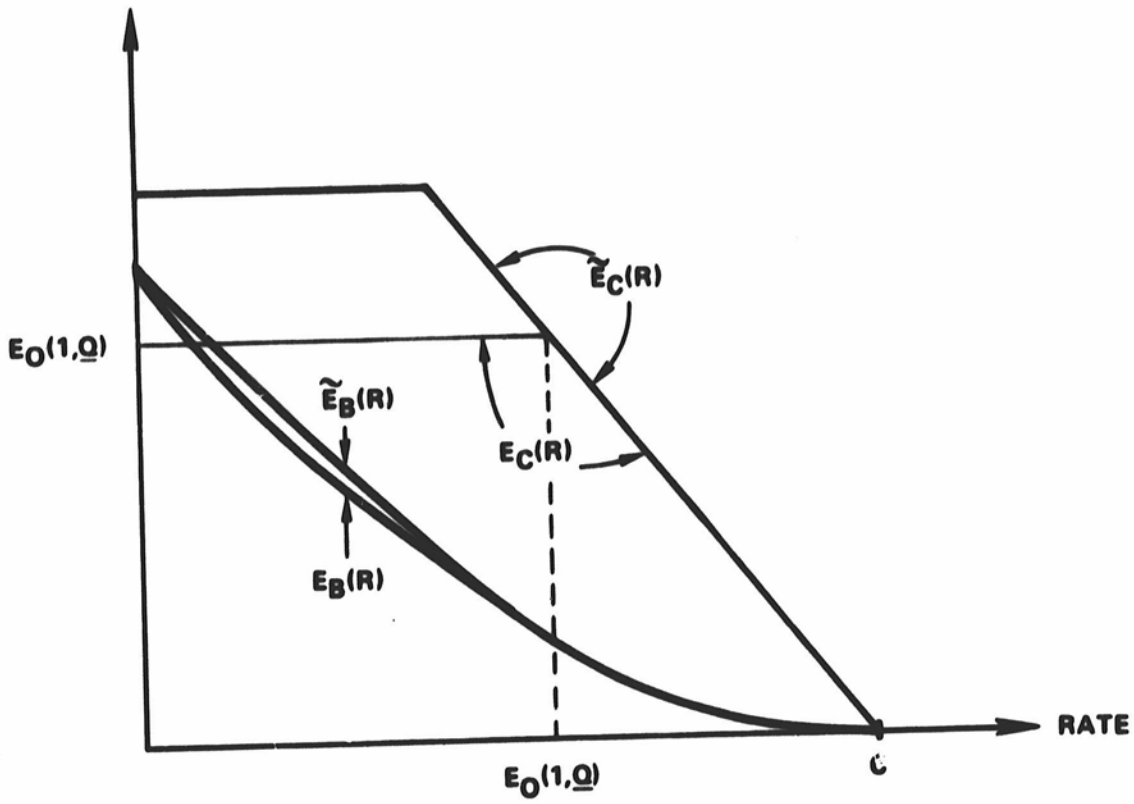


FIG. 3-2 BLOCK AND CONVOLUTIONAL RELIABILITY FUNCTIONS

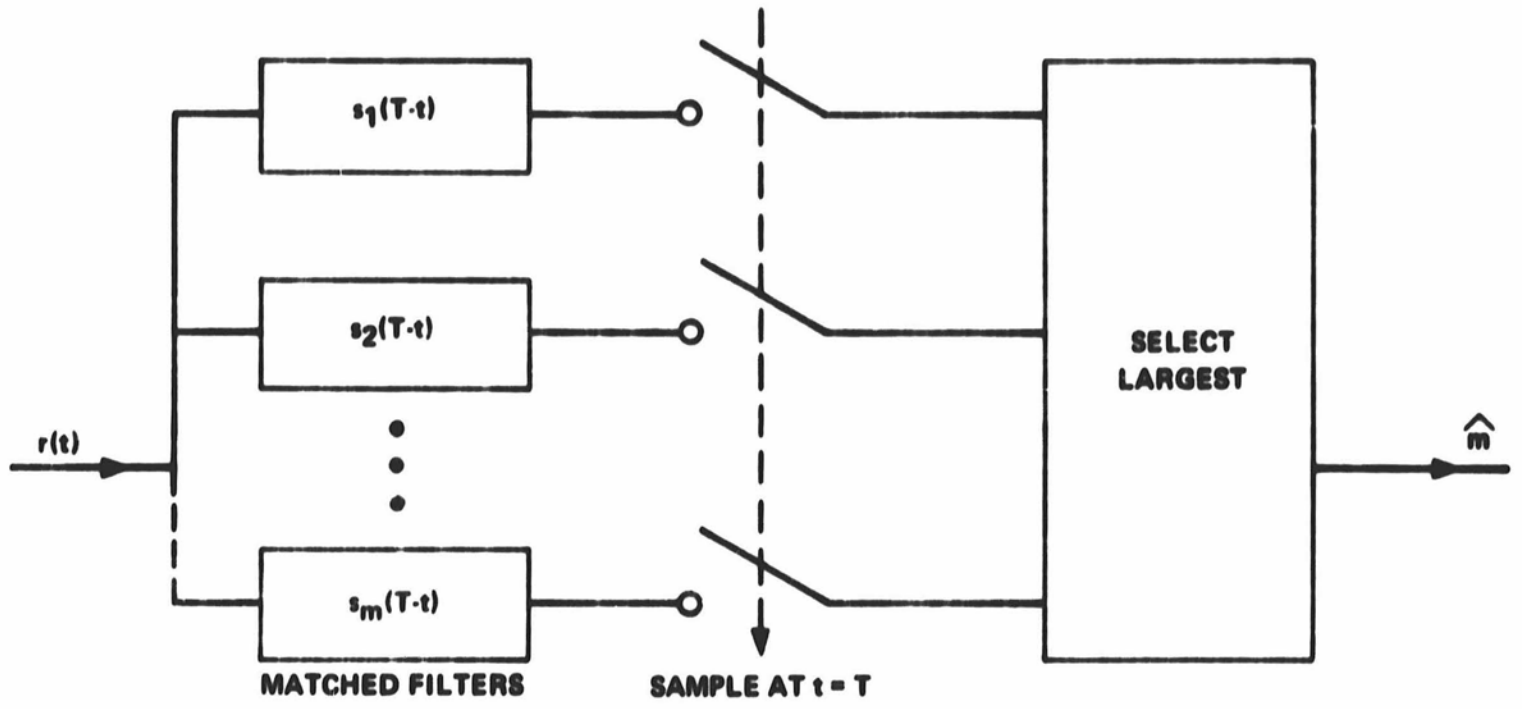


FIG. 4-1 OPTIMUM DETECTOR FOR WAVEFORM CODE

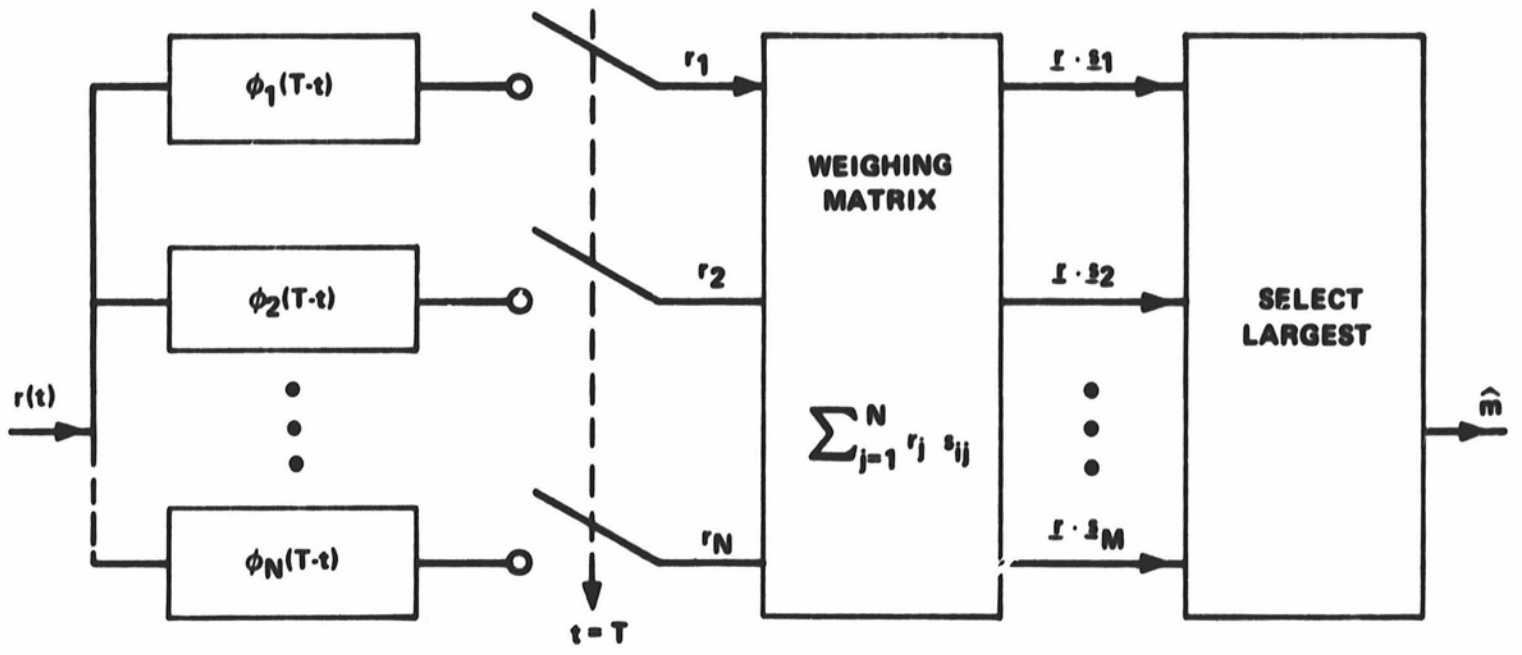


FIG. 4-2 ALTERNATE REALIZATION OF OPTIMUM DETECTOR FOR WAVEFORM CODE

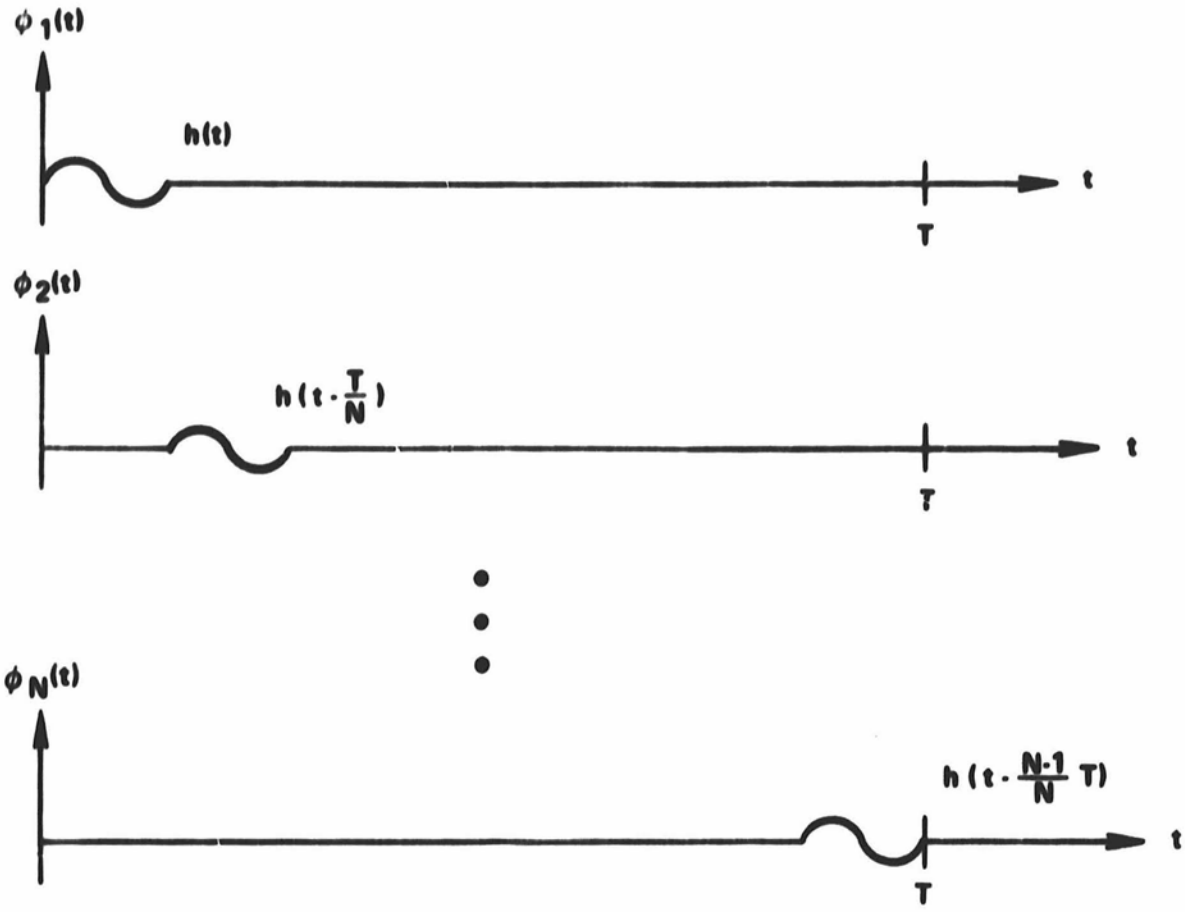
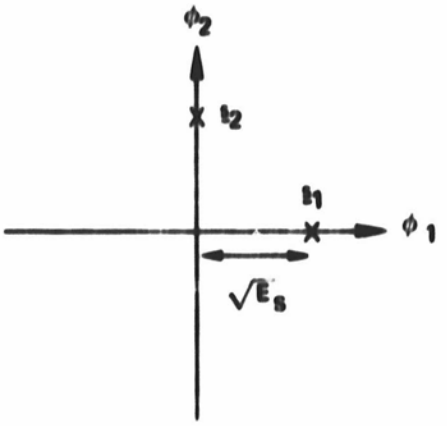
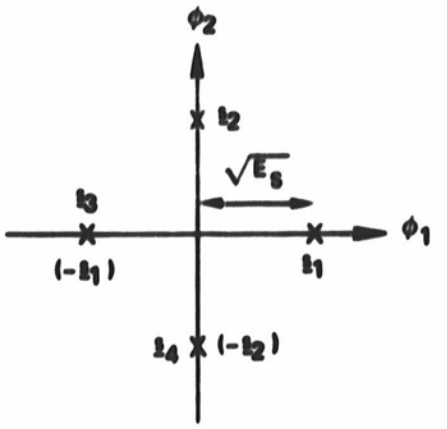


FIG. 4-3 TIME DISJOINT ORTHONORMAL WAVEFORMS

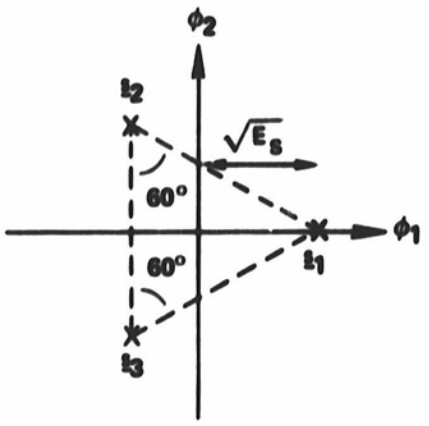
FIGURE 4 - 4 OPTIMUM RECEIVER FOR TIME DISJOINT PULSE WAVEFORMS



ORTHOGONAL CODE, N = M = 2



BIORTHOGONAL CODE, N = 2, M = 4



TRANSORTHOGONAL CODE, N= 2, M = 3

FIGURE 4 - 5 WAVEFORM CODES

$s_1 = [1\ 1\ 1\ 1]$
$s_2 = [1\ -1\ 1\ -1]$
$s_3 = [1\ 1\ -1\ -1]$
$s_4 = [1\ -1\ -1\ 1]$

ORTHOGONAL CODE

$s_1 = [1\ 1\ 1\ 1]$
$s_2 = [1\ -1\ 1\ -1]$
$s_3 = [1\ 1\ -1\ -1]$
$s_4 = [1\ -1\ -1\ 1]$
$s_5 = [-1\ -1\ -1\ -1]$
$s_6 = [-1\ 1\ -1\ 1]$
$s_7 = [-1\ -1\ 1\ 1]$
$s_8 = [-1\ 1\ 1\ -1]$

BIORTHOGONAL CODE

$s_1 = [1\ 1\ 1]$
$s_2 = [-1\ 1\ -1]$
$s_3 = [1\ -1\ -1]$
$s_4 = [-1\ -1\ 1]$

TRANSORTHOGONAL CODE

FIG. 4-6 CODES CONSTRUCTED FROM FOURTH ORDER HADAMARD MATRIX

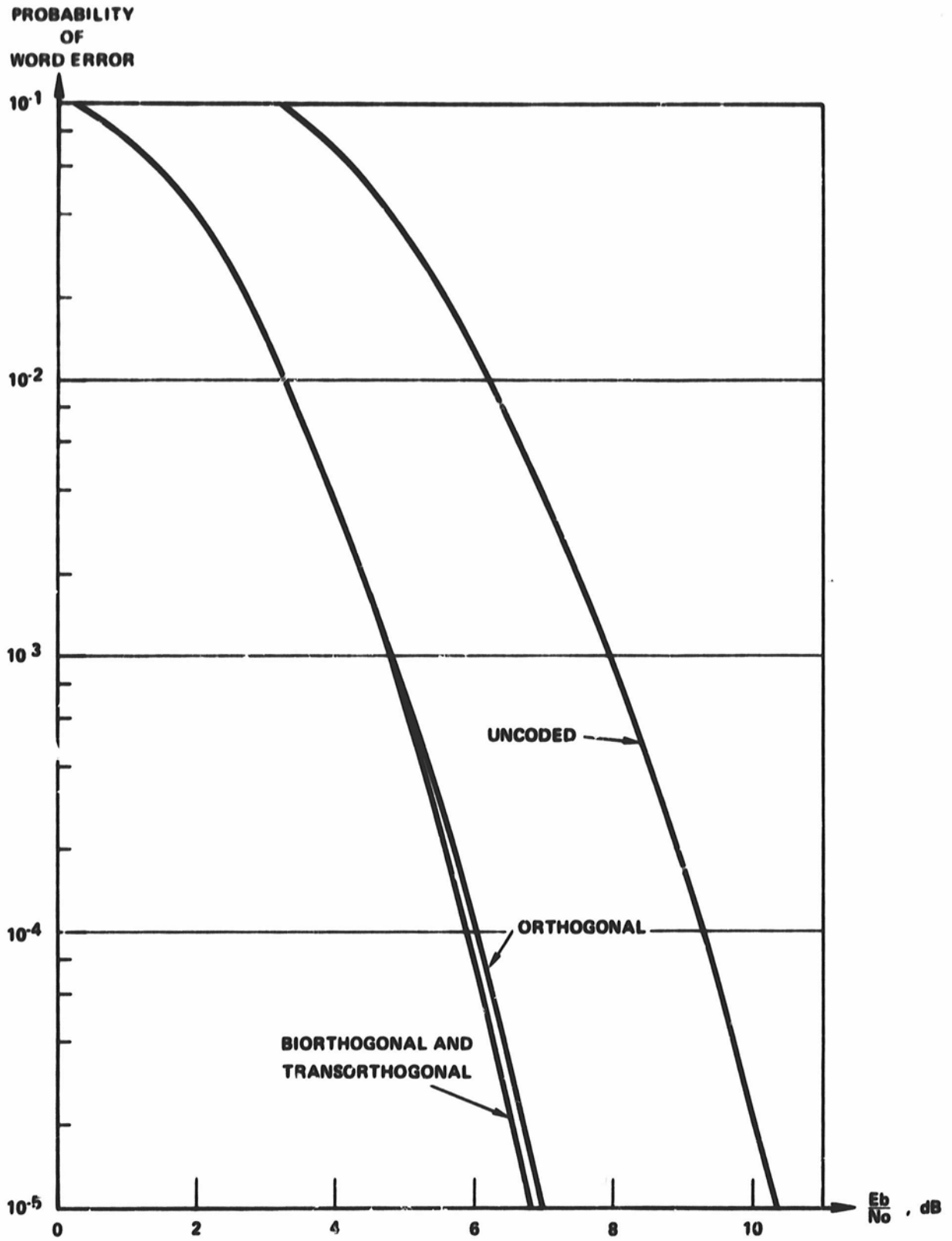


FIGURE 4 - 7 PROBABILITY OF WORD ERROR VS. $\frac{Eb}{No}$ FOR k = 5 WAVEFORM CODES

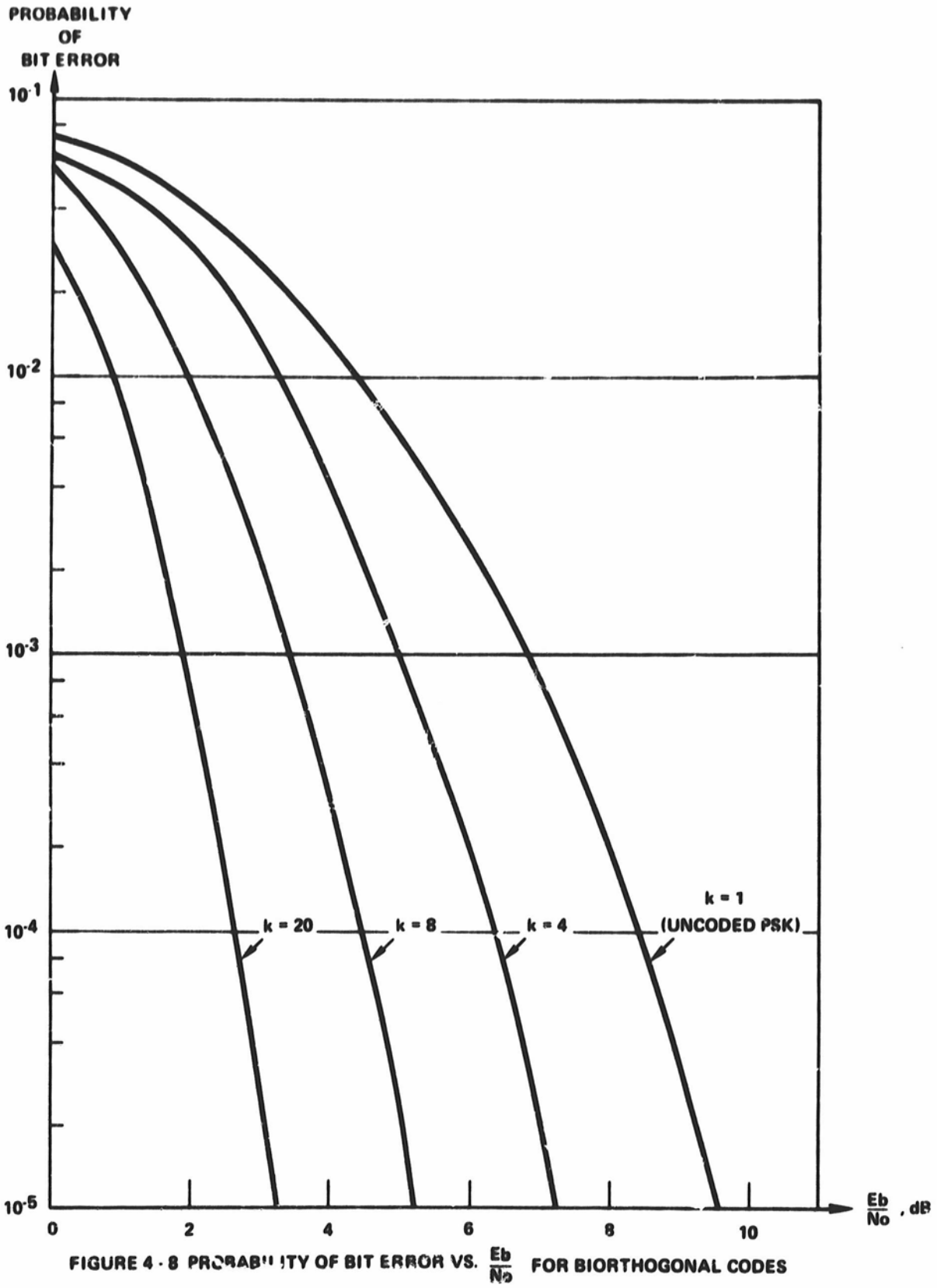


FIGURE 4 - 8 PROBABILITY OF BIT ERROR VS. $\frac{Eb}{No}$ FOR BIORTHOGONAL CODES

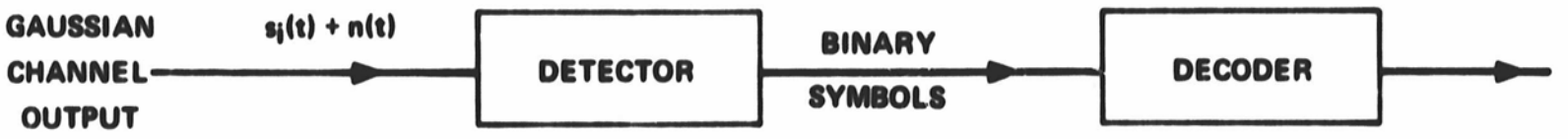


FIG. 4 - 9 DETECTOR FOR BCH CODE SYSTEM

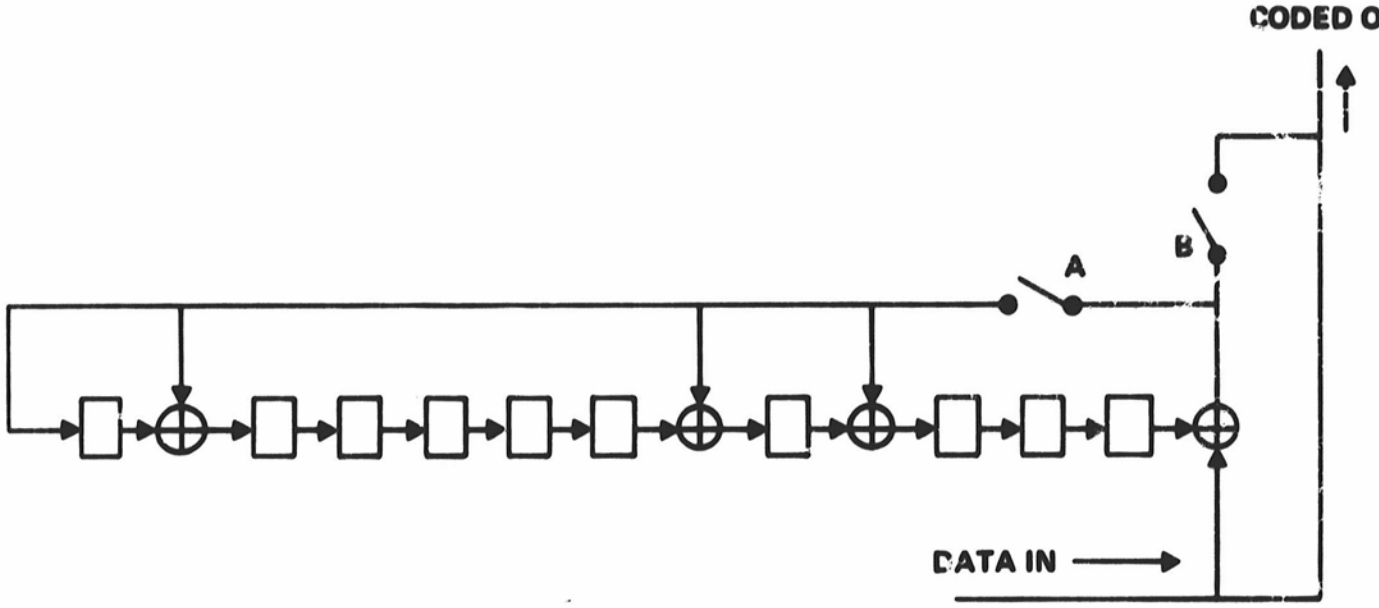


FIG. 4 - 10 (31, 21) BCH ENCODER

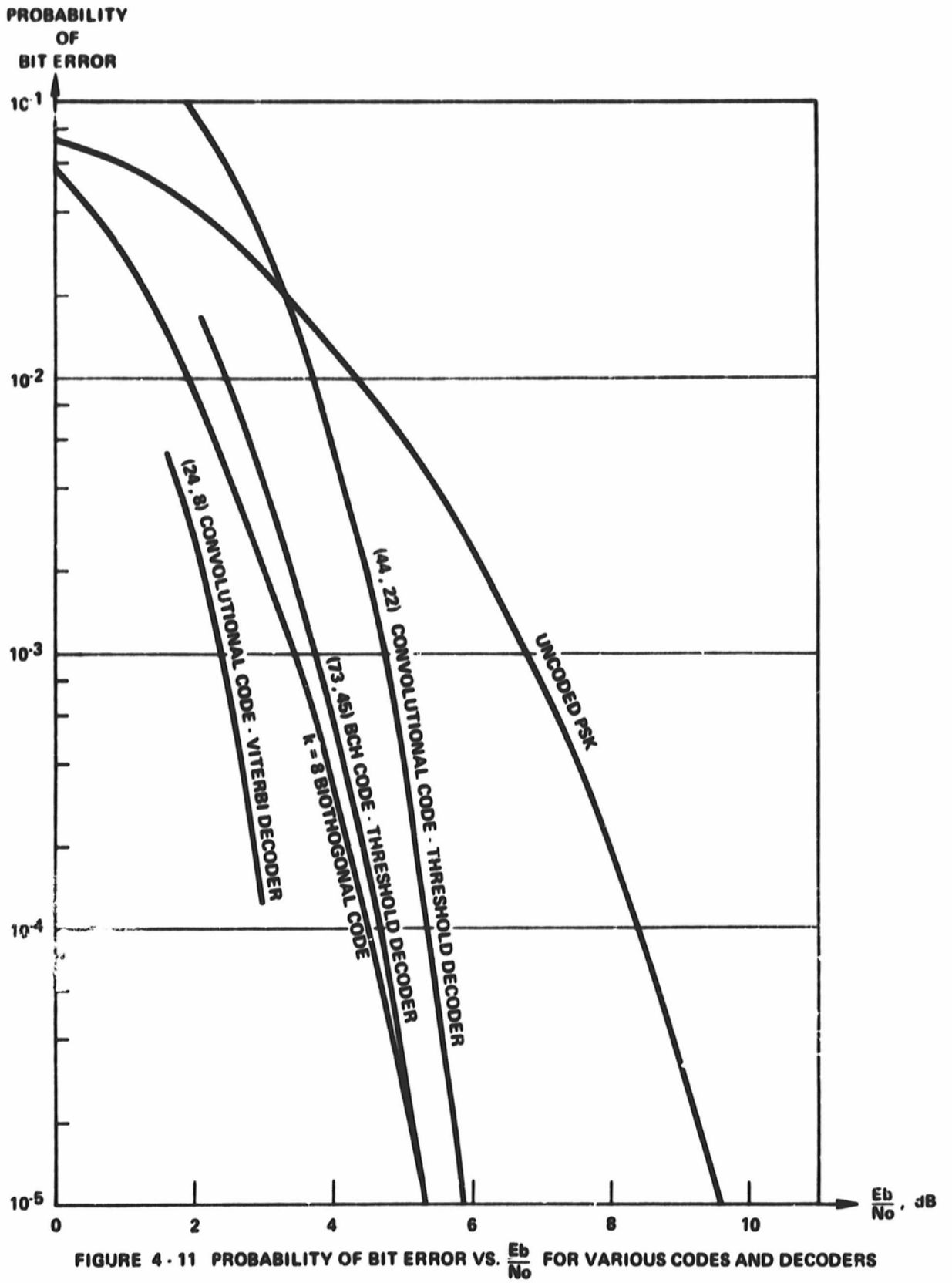


FIGURE 4 - 11 PROBABILITY OF BIT ERROR VS. $\frac{Eb}{No}$ FOR VARIOUS CODES AND DECODERS

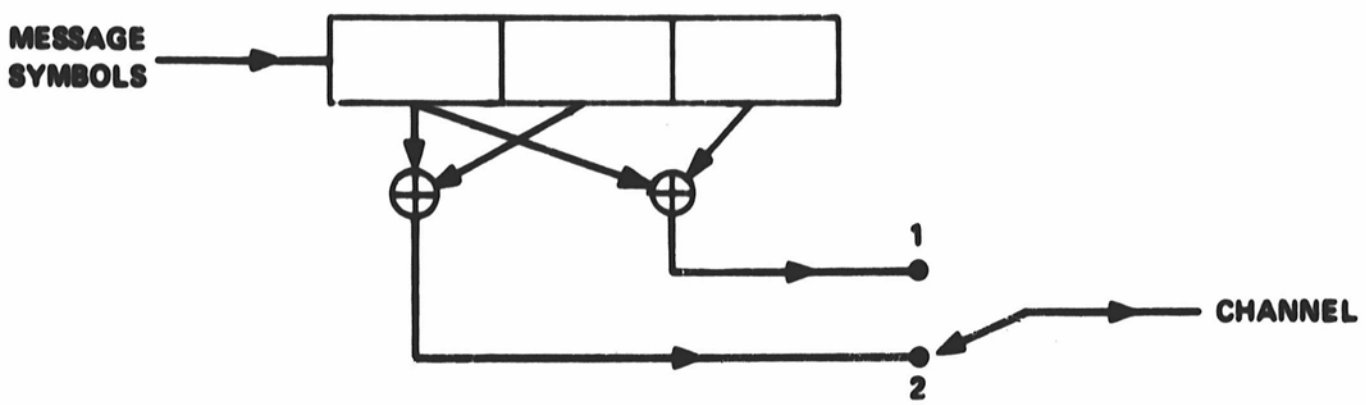


FIGURE 4 · 12 A SIMPLE CONVOLUTIONAL ENCODER

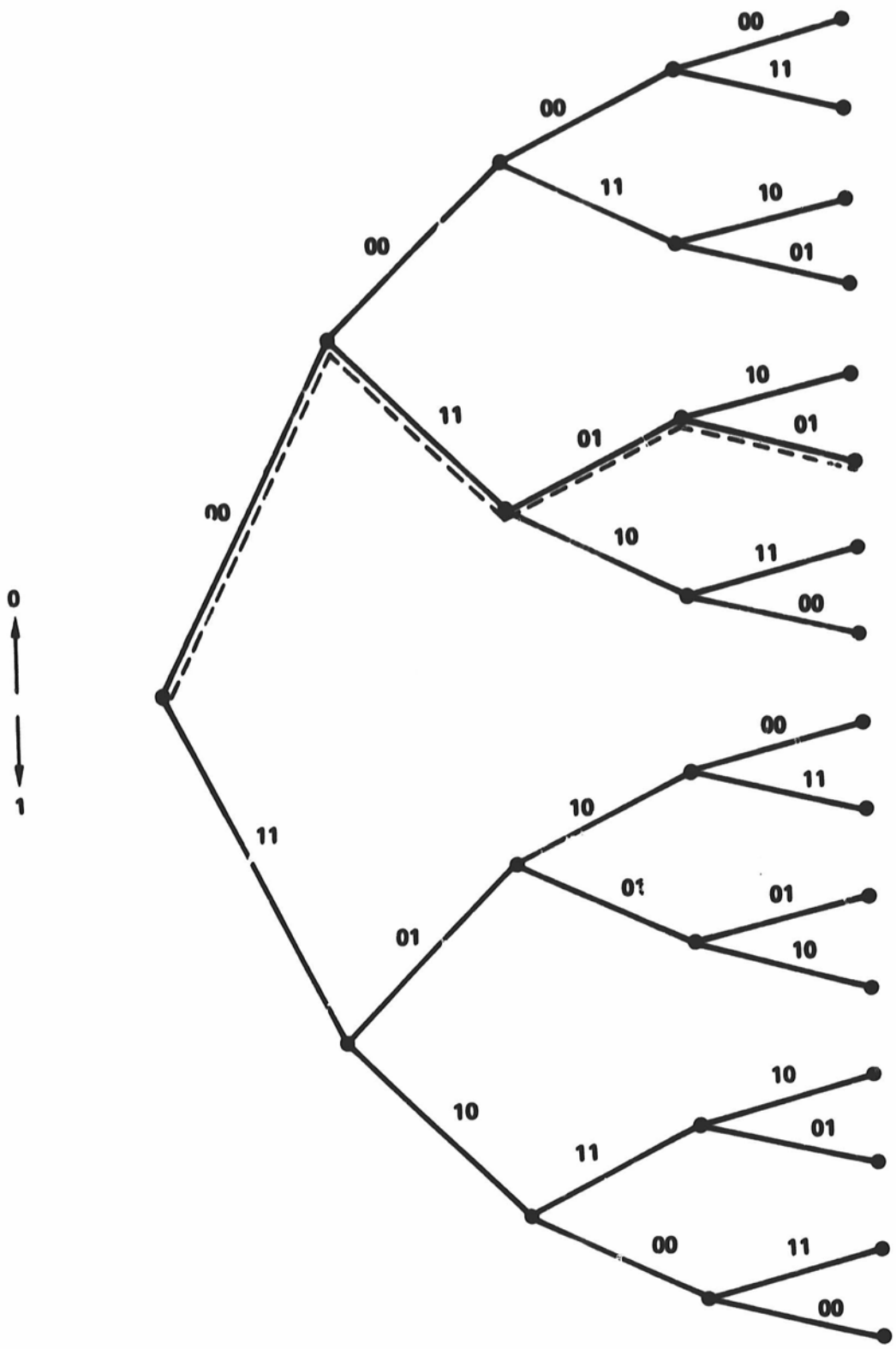


FIGURE 4 - 13 TREE STRUCTURE GENERATED BY ENCODER OF FIGURE 4 - 12

MESSAGE BIT	SHIFT REGISTER	OUTPUT
0	0 0 0	00
1	1 0 0	11
0	0 1 0	01
1	1 0 1	01

FIGURE 4 - 14 OPERATION OF CONVOLUTIONAL ENCODER

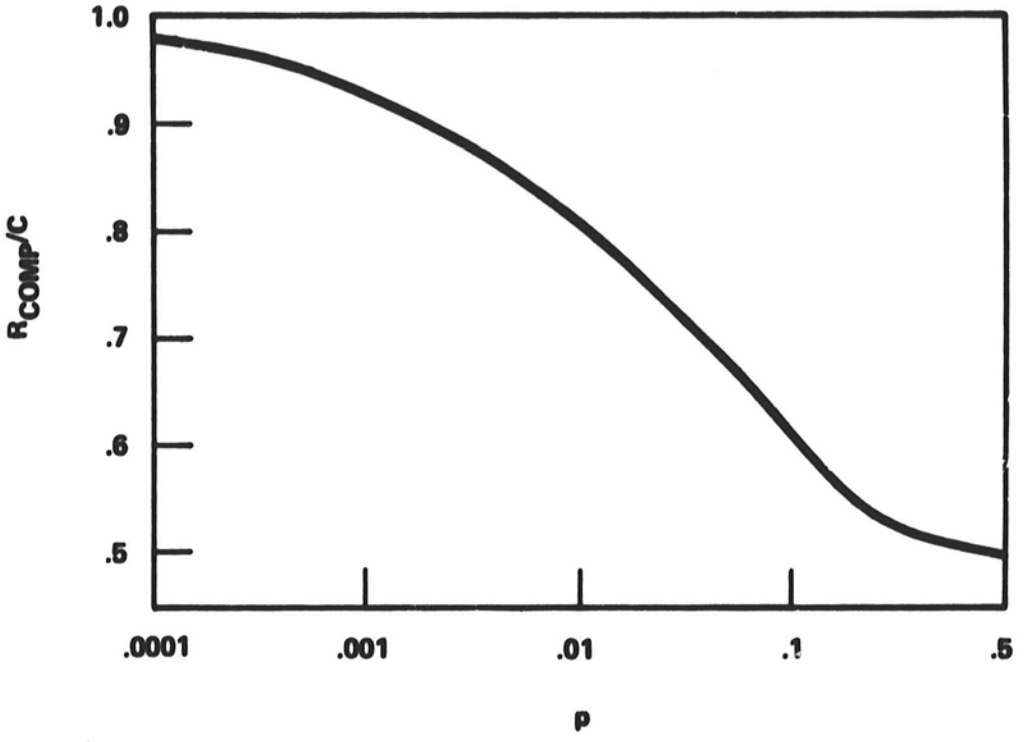


FIGURE 4 - 15 R_{COMP}/C AS A FUNCTION OF THE CROSSOVER PROBABILITY, p, OF A BINARY SYMMETRIC CHANNEL.

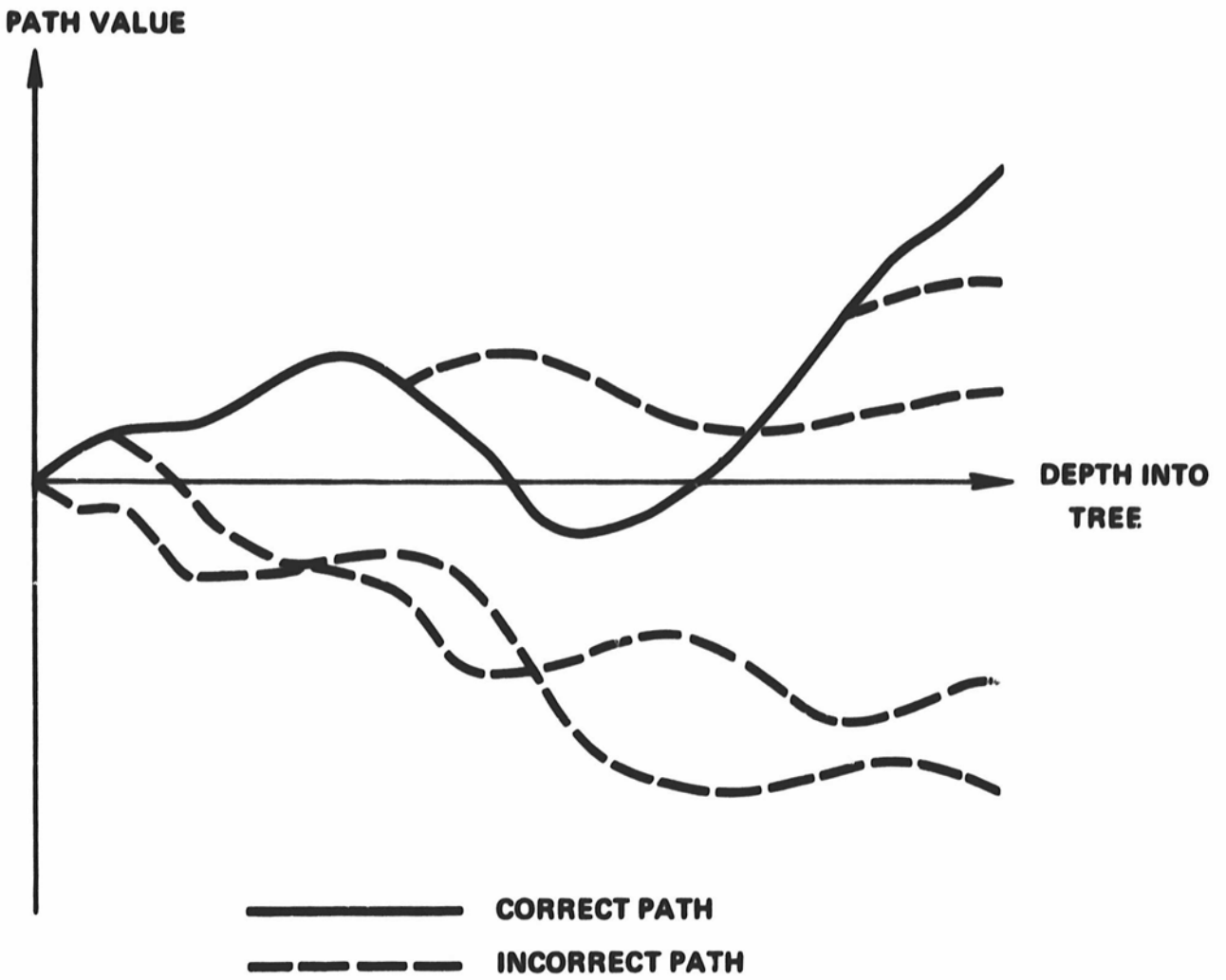


FIGURE 4 - 16 TYPICAL PATH VALUE TRAJECTORIES

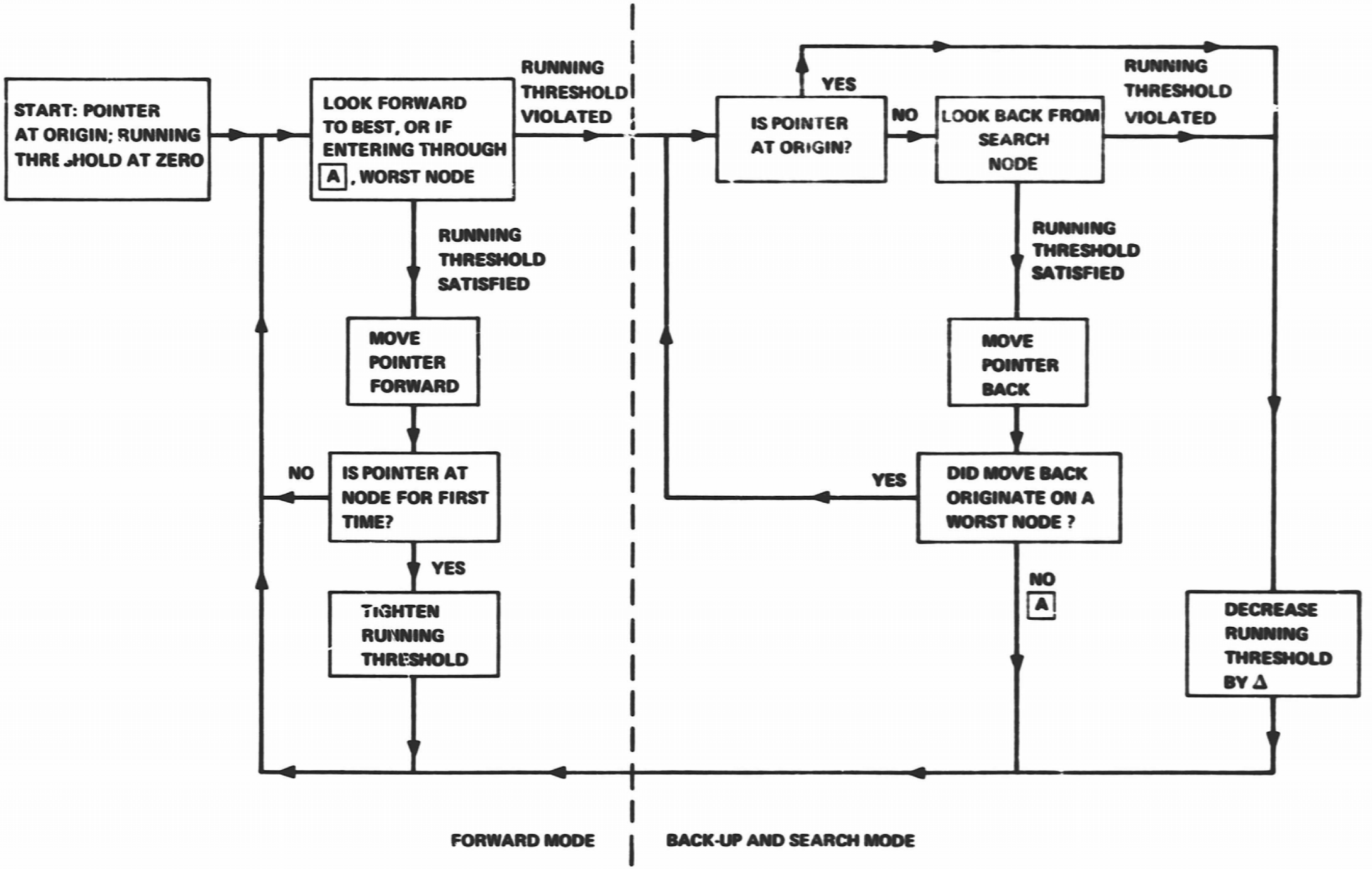


FIGURE 4 - 17 FLOW DIAGRAM OF FANO ALGORITHM

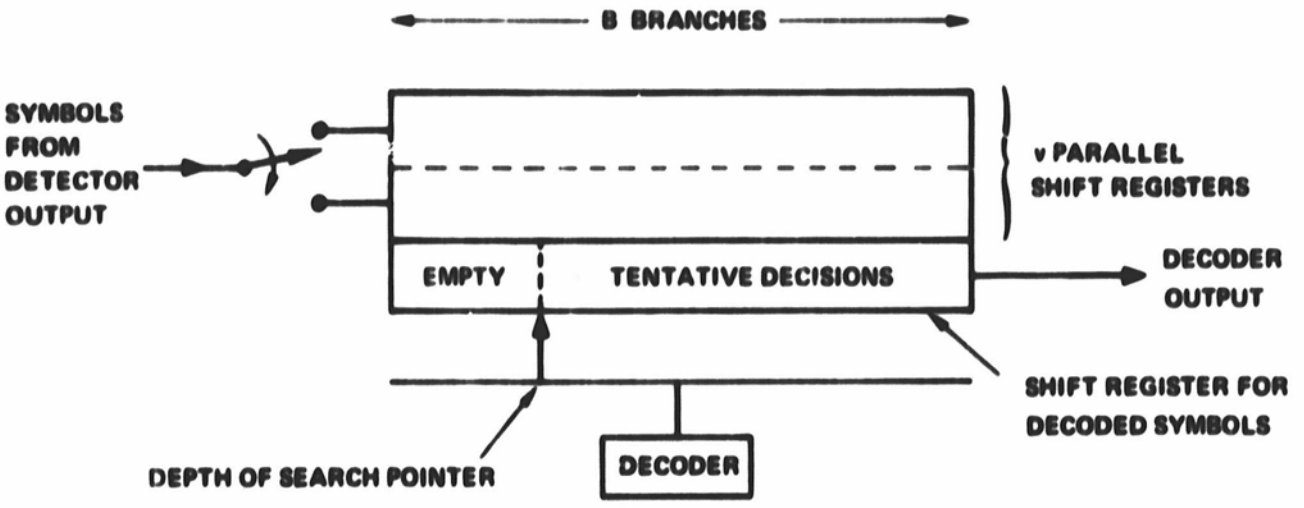


FIGURE 4 - 18 BUFFER FOR SEQUENTIAL DECODER

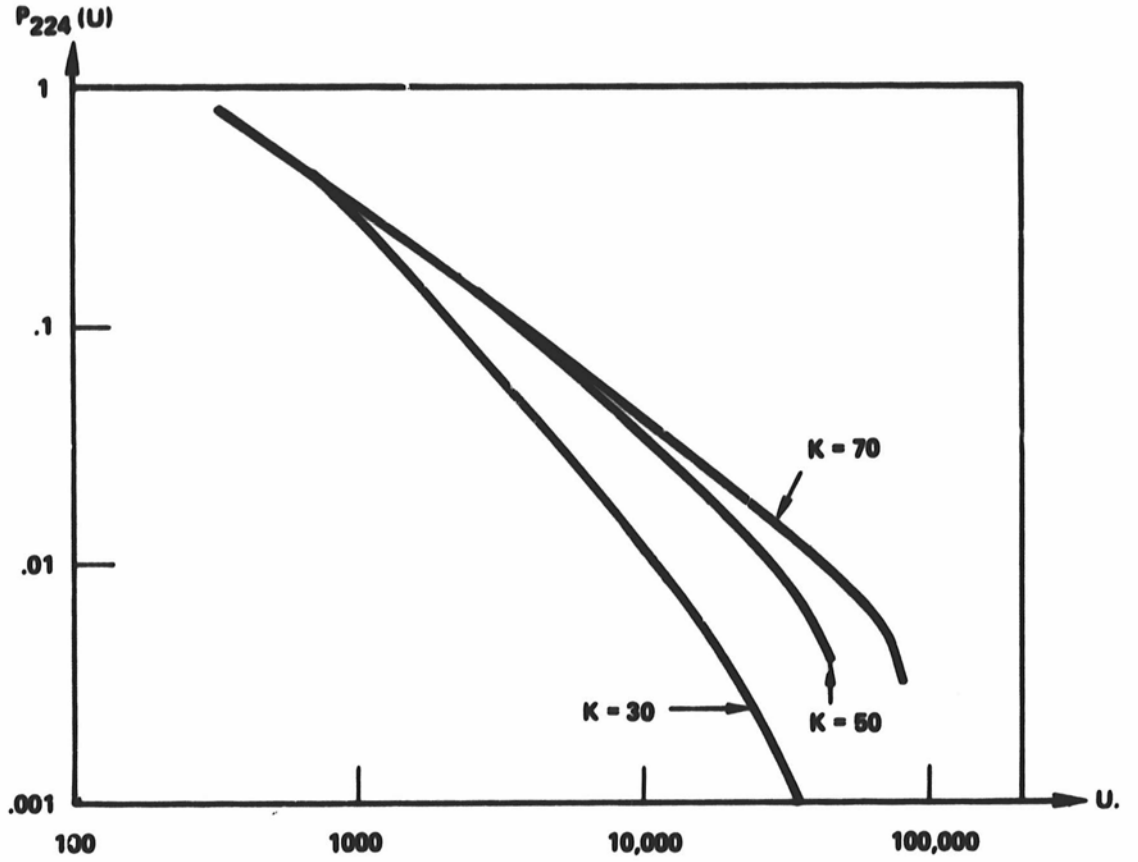


FIGURE 4 - 19 PROBABILITY THAT NUMBER OF COMPUTATIONS REQUIRED TO DECODE 224 BRANCHES EXCEEDS U.