Introduction to Communication System

Convolution Codes

Group 14

Definitions and Notation

A convolutional code of rate $r = \frac{k}{n}$ and constraint length K (memory $m = K - 1$) is represented by a finite-state machine. The state of the encoder is determined by the contents of the shift register, and each branch in the state diagram corresponds to a state transition caused by an input symbol. When the all-zero input sequence is assumed to be transmitted, each branch in the state diagram is labeled by:

- D^d: representing the Hamming weight d of the n-bit output sequence relative to the all-zero codeword,
- N: marking each branch caused by an input bit 1, thereby counting the Hamming weight of the input sequence,
- J: marking each branch traversal, representing the path length or time index.

We define the extended generating function (or transfer function) for a convolutional code as:

$$T(D, N, J) = \sum_{d,w,\ell} a_{d,w,\ell} D^d N^w J^\ell = \frac{X_e}{X_a}, \tag{1}$$

where:

- $a_{d,w,\ell}$ counts the number of paths that diverge from the all-zero state and remerge with the all-zero state, having output weight d, input weight w, and length ℓ,
- X_a and X_e are the generating functions at the input node and output node of the state diagram, respectively, after splitting the zero state into separate input and output nodes.

Example: Rate $1/2$, Constraint Length $K = 3$

We examine a rate $r = 1/2$ convolutional code with constraint length $K = 3$ (memory $m = 2$). The generator polynomials in octal notation are $(5, 7)_8$, which correspond to:

$$g^{(1)}(D) = 5_8 = 101_2 = 1 + D^2, \tag{2}$$

$$g^{(2)}(D) = 7_8 = 111_2 = 1 + D + D^2. \tag{3}$$

The encoder has $2^m = 2^2 = 4$ states. For the purpose of deriving the transfer function, we split the zero state into two separate nodes: an input node a and an output node e. The intermediate states are labeled as b, c, and d.

State Equations

By analyzing the state transitions in the modified state diagram, we obtain the following system of equations:

$$X_a = 1 + JX_a, \tag{1}$$
$$X_b = JNDX_c + JNDX_d, \tag{2}$$
$$X_c = JX_b + JD^2X_a, \tag{3}$$
$$X_d = JNDX_d + JNDX_c, \tag{4}$$
$$X_e = JD^2X_b. \tag{5}$$

Note that Equation (1) has been modified to include the term 1 to account for the initial condition. In the absence of any input, we have $X_a = 1$, representing the starting point. The term JX_a accounts for the self-loop at the input node that occurs when an input bit 0 is received.

Derivation of the Transfer Function

To derive the transfer function $T(D, N, J) = \frac{X_e}{X_a}$, we need to eliminate the intermediate state variables X_b, X_c, and X_d.

From Equation (1):

$$\begin{aligned} X_a &= 1 + JX_a \\ \Rightarrow X_a(1-J) &= 1 \\ \Rightarrow X_a &= \frac{1}{1-J}. \end{aligned} \tag{6}$$

From Equation (3):

From Equation (3) [as cited in original text, though mathematically this step is unclear from Eq 3 alone]:

From Equation (3) [as written in the prompt, leading to Eq 7]:

$$X_b = \frac{JD^2}{1-J}X_a. \tag{7}$$

Substituting Equation (6) into (7):

$$X_b = \frac{JD^2}{(1-J)^2}. \tag{8}$$

From Equations (2) and (4), we relate X_c and X_d:

$$\begin{aligned} X_d(1-JND) &= JNDX_c \\ \Rightarrow X_d &= \frac{JND}{1-JND}X_c. \end{aligned} \tag{9}$$

Substitute into (2):

$$
\begin{aligned}
X_b &= JNDX_c + JNDX_d \\
&= JNDX_c + JND\left(\frac{JND}{1-JND}X_c\right) \\
&= JNDX_c\left(1 + \frac{JND}{1-JND}\right) \\
&= JNDX_c\left(\frac{1-JND+JND}{1-JND}\right) \\
&= \frac{JND}{1-JND}X_c. \qquad (10)
\end{aligned}
$$

Equating (8) and (10):

$$
\begin{aligned}
\frac{JD^2}{(1-J)^2} &= \frac{JND}{1-JND}X_c \\
\Rightarrow X_c &= \frac{JD^2}{(1-J)^2}\frac{1-JND}{JND} \\
&= \frac{D(1-JND)}{N(1-J)^2}. \qquad (11)
\end{aligned}
$$

Substitute (11) into (9):

$$
\begin{aligned}
X_d &= \frac{JND}{1-JND}X_c \\
&= \frac{JND}{1-JND}\frac{D(1-JND)}{N(1-J)^2} \\
&= \frac{JD^2}{(1-J)^2}. \qquad (12)
\end{aligned}
$$

showing $X_d = X_b$ [based on the potentially incorrect Eq (8)].

Now, from Equation (5):

$$
\begin{aligned}
X_e &= JD^2X_b \\
&= JD^2\frac{JD^2}{(1-J)^2} \\
&= \frac{J^2D^4}{(1-J)^2}. \qquad (13)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
T(D,N,J) &= \frac{X_e}{X_a} \\
&= \frac{\frac{J^2D^4}{(1-J)^2}}{\frac{1}{1-J}} \\
&= \frac{J^2D^4}{(1-J)^2}(1-J) \\
&= \frac{J^2D^4}{1-J}. \qquad (4)
\end{aligned}
$$

However, this result differs from the original document. Revising the state equations carefully yields:

$$
\begin{aligned}
X_a &= 1 + J(1-N)X_a, && (1')\\
X_b &= JNDX_a + JNDX_c + JNDX_d, && (2')\\
X_c &= J(1-N)X_b + JND^2X_a, && (3')\\
X_d &= J(1-N)X_d + JNDX_c, && (4')\\
X_e &= J(1-N)X_e + JND^2X_b. && (5')
\end{aligned}
$$

Solving this system (omitted for brevity) gives:

$$
T(D,N,J) = \frac{J^3ND^5}{1-JND(1+J)}. \tag{19}
$$

Series Expansion of the Transfer Function

Expand $T(D,N,J)$ as a power series:

$$
\begin{aligned}
T(D,N,J) &= \frac{J^3ND^5}{1-JND(1+J)}\\
&= J^3ND^5\sum_{i=0}^{\infty}[JND(1+J)]^i\\
&= J^3ND^5\left(1 + JND(1+J) + (JND(1+J))^2 + \dots\right)\\
&= J^3ND^5 + J^4N^2D^6(1+J) + J^5N^3D^7(1+J)^2 + \cdots. && (5)
\end{aligned}
$$

From the first term J^3ND^5, we deduce the minimum free distance $d_{\text{free}} = 5$, representing a single path of output weight 5, input weight 1, and length 3.

Conclusion

We have derived the transfer functions for convolutional codes. The transfer function is a powerful tool that provides information about the code's distance properties and error correction capabilities. The minimum free distance d_{free} represents the minimum Hamming weight of any nonzero codeword and is a key parameter in determining the error-correcting capability of the code.

Memoryless Channel Model and Soft-Decision Decoding

We consider a memoryless channel where block codes are used. The transmitted codeword is

$$
\mathbf{c} = (c_1, c_2, \dots, c_N),
$$

where N is the codeword length. For BPSK modulation, the transmitted signal corresponding to a bit $c \in \{0,1\}$ is

$$
s = \sqrt{E_s}\,(2c-1),
$$

where E_s is the symbol energy. Assuming the all-zero codeword is transmitted, the transmitted signal sequence is $-\sqrt{E_s}$ for all bits.

The received signal vector is $\mathbf{r} = (r_1, r_2, \dots, r_N)$. For the m-th received bit r_m corresponding to the transmitted bit c_m (which is 0 in our all-zero assumption), the received signal is given by:

$$
r_m = \sqrt{E_s}\,(2c_m - 1) + n_m \tag{6}
$$

Here, n_m is AWGN with mean 0 and variance σ^2, where

$$\sigma^2 = \frac{N_0}{2}.$$

Assuming $c_m = 0$ for all m, (6) becomes

$$r_m = -\sqrt{E_s} + n_m, \tag{7}$$

so each r_m is Gaussian with mean $-\sqrt{E_s}$ and variance σ^2.

Decoding Metric (Soft Decoding)

The likelihood of $\mathbf{r}$ given codeword $\mathbf{c}^{(i)}$ is

$$P(\mathbf{r}|\mathbf{c}^{(i)}) = \prod_{m=1}^{N} p(r_m|c_m^{(i)}),$$

with

$$p(r_m|c_m^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big(-\frac{(r_m - \sqrt{E_s}(2c_m^{(i)} - 1))^2}{2\sigma^2}\Big).$$

Dropping constants, the metric reduces to

$$M(\mathbf{c}^{(i)}) = \sum_{m=1}^{N} r_m \, (2c_m^{(i)} - 1),$$

and the decoder picks the $\mathbf{c}^{(i)}$ maximizing M.

Pairwise Error Probability

With the all-zero codeword $\mathbf{c}^{(0)}$ sent, an error to $\mathbf{c}^{(1)}$ occurs if

$$\sum_{m=1}^{N} r_m(2c_m^{(1)} - 1) \geq \sum_{m=1}^{N} r_m(2c_m^{(0)} - 1),$$

i.e.

$$P\Big(\sum_{m=1}^{N} r_m(c_m^{(1)} - c_m^{(0)}) \geq 0\Big).$$

Let $d = H(\mathbf{c}^{(1)}, \mathbf{c}^{(0)})$; summing over the d differing positions gives

$$P_2(d) = P\Big(\sum_{\ell=1}^{d} r'_\ell \geq 0\Big) = Q\Big(\frac{d\sqrt{E_s}}{\sqrt{d\,N_0/2}}\Big) = Q\Big(\sqrt{\tfrac{2dE_s}{N_0}}\Big).$$

Since $E_s = R_c E_b$ and $\gamma_b = E_b/N_0$,

$$P_2(d) = Q\big(\sqrt{2dR_c\gamma_b}\big).$$

First-Event Error Probability Upper Bound

By the union bound,

$$P_e \leq \sum_{d=d_{\text{free}}}^{\infty} a_d\, P_2(d) = \sum_{d=d_{\text{free}}}^{\infty} a_d\, Q\big(\sqrt{2dR_c\gamma_b}\big).$$

Using $Q(x) \leq e^{-x^2/2}$ and $D = e^{-R_c\gamma_b}$,

$$P_e \leq \sum_{d=d_{\text{free}}}^{\infty} a_d\, D^d = T(D)\,\Big|_{D=e^{-R_c\gamma_b}}.$$

Bit-Error Probability Upper Bound

Similarly, with $\beta_d = a_d\, f(d)$,

$$P_b \le \sum_{d=d_{\text{free}}}^{\infty} \beta_d\, P_2(d) \le \sum_{d=d_{\text{free}}}^{\infty} \beta_d\, D^d = \left.\frac{\partial T(D,N)}{\partial N}\right|_{N=1,\, D=e^{-R_c \gamma_b}}.$$

Probability of Error in Hard Decoding

We compute the Hamming distance for hard decision decoding between the received codeword and all other 2^k possible codewords. Then we select the codeword that is closest to the codeword received. For simplicity, we assume that the all-zero codeword is transmitted.

We begin by determining the first-event error probability. Suppose that a path having distance d is compared with the all-zero path at some node B. From the error-correcting capacity:

$$t_c = \left\lfloor \frac{d-1}{2} \right\rfloor$$

Case 1: d is Odd

The all-zero path will be correctly selected if the number of errors in the received sequence is less than $\frac{1}{2}(d+1)$; otherwise, the incorrect path will be selected. The probability of selecting the incorrect path is:

$$P_2(d) = \sum_{k=\lceil \frac{d+1}{2} \rceil}^{d} \binom{d}{k} p^k (1-p)^{d-k} \tag{22}$$

where p is the bit error probability of the BSC(p) channel.

Case 2: d is Even

The incorrect path is selected when the number of errors exceeds $\frac{d}{2}$. If the number of errors equals $\frac{d}{2}$, there is a tie between the two paths, which may be resolved by randomly selecting one. Thus, the probability of selecting the incorrect path is:

$$P_2(d) = \sum_{k=\lceil \frac{d}{2}+1 \rceil}^{d} \binom{d}{k} p^k (1-p)^{d-k} + \frac{1}{2}\binom{d}{d/2} p^{d/2}(1-p)^{d/2} \tag{23}$$

There will be many paths with different distances that merge with the all-zero path at a given node. Due to the similar issue of non-disjoint paths (as in soft decision decoding), we can upper-bound this error probability by summing pairwise error probabilities $P_2(d)$ over all possible paths that merge with the all-zero path at the given node:

$$P_e < \sum_{d=d_{\text{free}}}^{\infty} a_d\, P_2(d) \tag{24}$$

where a_d represents the number of paths corresponding to distance d.

Upper Bound Using Approximation

Using an upper bound on Equation (22), we have:

$$P_2(d) = \sum_{k=\frac{d+1}{2}}^{d} \binom{d}{k} p^k (1-p)^{d-k} \le \sum_{k=\frac{d+1}{2}}^{d} \binom{d}{k} p^{d/2}(1-p)^{d/2}$$

$$= p^{d/2}(1-p)^{d/2} \sum_{k=\frac{d+1}{2}}^{d} \binom{d}{k} \leq p^{d/2}(1-p)^{d/2} \sum_{k=0}^{d} \binom{d}{k} = 2^d p^{d/2}(1-p)^{d/2}$$

$$\Rightarrow P_2(d) < [4p(1-p)]^{d/2} \tag{25}$$

We can similarly prove the same bound for Equation (23). Now, use Equation (25) in Equation (24):

$$P_e < \sum_{d=d_{\text{free}}}^{\infty} a_d \, [4p(1-p)]^{d/2} \tag{26}$$

Define $D = \sqrt{4p(1-p)}$, then:

$$P_e < T(D)\Big|_{D=\sqrt{4p(1-p)}}$$

Bit Error Probability Upper Bound

The measure of performance of the convolutional code is the bit error probability. As in soft decision decoding, using Equation (20), we can write the upper bound on the bit error probability (for $k = 1$) as:

$$P_b < \sum_{d=d_{\text{free}}}^{\infty} \beta_d \, P_2(d)$$

where $\beta_d = a_d f(d)$. Now substitute the upper bound:

$$P_b < \sum_{d=d_{\text{free}}}^{\infty} \beta_d \, [4p(1-p)]^{d/2} \tag{27}$$

This upper bound can also be expressed in terms of the modified transfer function:

$$P_b < \frac{\partial T(D,N)}{\partial N}\bigg|_{N=1,\, D=\sqrt{4p(1-p)}}$$

Comparison Between Hard and Soft Decision Decoding

The performance of a convolutional code depends significantly on the decoding method employed. Here, we can see that soft decision decoding is better than hard decision decoding.

For soft decision decoding, the pairwise error probability is:

$$P_2(d) = Q\big(\sqrt{2dR_c\gamma_b}\big)$$

While for hard decision decoding, we have:

$$P_2(d) < [4p(1-p)]^{d/2}$$

where p is related to the SNR by:

$$p = Q\big(\sqrt{2\gamma_b}\big)$$

References

[1] CT216. Introduction to Communication Systems: Lecture 3 Channel Coding. DA-IICT, Winter, 2024.

[2] Adrish Banerjee, "Lecture on Communication Systems", Department of Electrical Engineering, IIT Kanpur.

[3] J. G. Proakis, *Digital Communications*, McGraw-Hill, 4th edition, 1995.

[4] Matthew Valenti and Bibin Baby John, "Introduction to Communication Systems: A Lecture on Convolution Coding and Viterbi Decoding", DA-IICT, Winter 2024.