Dhirubhai Ambani Institute of Information and Communication Technology

CT 216: Introduction to Communication Systems

Convolutional Codes Analysis

Professor: Yash Vasavada

Mentor: Heman Chauhan

Group 10

202301103	Krish Makavana
202301106	Kaushal Kanani
202301110	Dhruv Gohil
202301102	Het Ladani
202301105	Ved Ambani
202301107	Shambhavi Singh
202301101	Pranav Mandani
202301104	Harshil Vora

1.Introduction

Convolutional codes are a class of error-correcting codes used extensively in digital communication systems for their ability to enhance coding gain. Unlike block codes, they process data as a continuous stream, encoding each bit or small group of bits based on both the current input and a set number of prior inputs, defined by the code's memory or constraint length.

Decoding these codes often relies on the Viterbi algorithm, which employs maximum likelihood sequence estimation (MLSE). This method evaluates the code's trellis structure to determine the most probable sequence of transmitted bits given the received data.

This document explores the error performance of convolutional codes when decoded with the Viterbi algorithm. It establishes upper bounds on error probability for both soft-decision and hard-decision decoding, emphasizing the influence of factors like signal-to-noise ratio (SNR), code rate (Rc), constraint length (Kc), and the code's transfer function.

2.Viterbi Algorithm: Analysis

The **Viterbi algorithm** is a dynamic programming technique used to determine the most probable sequence of hidden states (called the *Viterbi path*) in a Hidden Markov Model (HMM), given a sequence of observations.

Time Complexity

For a sequence of length T and N states in the HMM:

$$\text{Time Complexity} = \mathcal{O}(N^2 \cdot T)$$

Each state at time t considers transitions from all N states at time $t-1$, repeated over T time steps.

Space Complexity

$$\text{Space Complexity} = \mathcal{O}(N \cdot T)$$

This is due to storing the path probabilities and backpointers for all states at each time step.

Key Equations

Let:

- $V_t(j)$: Highest probability of a state sequence ending in state j at time t
- a_{ij}: Transition probability from state i to state j
- $b_j(o_t)$: Observation likelihood of o_t in state j
- π_j: Initial probability of state j

Initialization:

$$V_1(j) = \pi_j \cdot b_j(o_1)$$

Recursion:

$$V_t(j) = \max_i \left[V_{t-1}(i) \cdot a_{ij}\right] \cdot b_j(o_t)$$

Termination:

$$\max_j V_T(j)$$

Backtracking: Used to retrieve the most likely state sequence by tracing back the path pointers stored during recursion.

3.Soft Decision Decoding Analysis

Equation (8-2-20): Pairwise Error Probability $P_2(d)$

Derivation

The soft decision decoder computes a metric for each path in the trellis known as the Path Metric.

$$CM^{(i)} = \sum_j r_{jm}(2c_{jm'} - 1) \tag{8-2-16}$$

- c_{jm}: Coded symbol related to the m^{th} symbol of the j^{th} branch. Its components are c_{jm1}, c_{jm2}, etc.
- r_{jm}: Output from the demodulator, i.e., the Viterbi decoder inputs corresponding to the m^{th} symbol of the j^{th} branch in the trellis where,

$$r_{jm} = \sqrt{\varepsilon_c}(2c_{jm} - 1) + \eta_{jm}$$

- η_{jm}: Represents the additive white Gaussian noise that affects the reception of the m^{th} symbol of the j^{th} branch in the trellis.

Thus, for an all-zero path:

$$CM^{(0)} = \sqrt{\varepsilon_c}Bn - \sum_{j=1}^{B}\sum_{m=1}^{n}(\eta_{jm}) \qquad (8\text{-}2\text{-}17)$$

- Bn: Total number of transmitted coded symbols.
- B: Number of trellis branches.
- n: Number of coded symbols per branch.

Suppose the incorrect path (say $i = 1$) differs from the all-zero path in d bits:

$$CM^{(1)} = \sum_{j=1}^{B} \sum_{m=1}^{n} r_{jm} \left(2c_{jm}^{(1)} - 1\right)$$

The decoder compares the all-zero path metric $CM^{(0)}$ with that of the incorrect path $CM^{(1)}$:

$$P_2(d) = P\left(CM^{(1)} \geq CM^{(0)}\right) = P\left(CM^{(1)} - CM^{(0)} \geq 0\right)$$

$$P_2(d) = P\left(2\sum_{j=1}^{B} \sum_{m=1}^{n} r_{jm}(c_{jm}^{(1)} - c_{jm}^{(0)}) \geq 0\right) \qquad (8\text{-}2\text{-}18)$$

Since the sequences in the two paths are identical except in the d positions, the above equation can also be written as:

$$P_2(d) = P\left(\sum_{l=1}^{d} r_l' \geq 0\right) \qquad (8\text{-}2\text{-}19)$$

where $\{r_l'\}$ is a set representing the input to the decoder for these d bits.

The $\{r_l'\}$ are independent and identically distributed normal random variables with:

$$\text{mean} = -\sqrt{\varepsilon_c}, \quad \text{variance} = \frac{1}{2}N_0$$

Therefore, the error probability in the pairwise comparison of two paths that differ by d bits is:

$$P_2(d) = Q\left(\sqrt{\frac{2d\varepsilon_c}{N_0}}\right)$$

$$P_2(d) = Q\left(\sqrt{2\gamma_b R_c d}\right) \qquad (8\text{-}2\text{-}20)$$

Explanation

- $P_2(d)$: The pairwise error probability as a function of distance d.
- $Q(x)$: Tail probability of the Gaussian distribution.
- $\gamma_b = \frac{E_b}{N_0}$: Bit Signal-to-Noise Ratio (SNR).
- $R_c = \frac{k}{n}$: Code rate.
- d: Hamming distance between paths.
- ε_c: The energy per coded bit.
- N_0: The one-sided power spectral density of the additive white Gaussian noise (AWGN).

The higher the SNR or the greater the distance d, the less likely the decoder will choose the wrong path.

Equation (8-2-21): First-Event Error Probability P_e

Derivation

We have derived the error probability for a path of distance d, but we can also sum this error probability over all possible path distances. As there are many possible paths with different distances that merge with the all-zero path at a particular given node, this summation will give us an upper bound on the first-event error probability:

$$P_e \leq \sum_{d=d_{\text{free}}}^{\infty} a_d P_2(d) \leq \sum_{d=d_{\text{free}}}^{\infty} a_d Q\left(\sqrt{2\gamma_b R_c d}\right) \qquad \text{(8-2-21)}$$

Explanation

- a_d: Number of paths that differ from the all-zero path at d positions and merge with the all-zero path for the first time.
- d_{free}: Minimum distance between codewords (free distance).

This equation provides an upper bound on the probability that an incorrect path merges with the correct path at a specific node. The term $a_d P_2(d)$ represents the contribution of each error event and its associated probability.

Equation (8-2-21) specifically bounds the first-event error probability, acknowledging that the events contributing to $\{P_2(d)\}$ are not mutually exclusive. By summing over all $d \geq d_{\text{free}}$, it is implicitly assumed that the convolutional code has infinite length.

Equation (8-2-26): Bit Error Probability P_b

Derivation

A more useful measure of performance of a convolutional code is the bit error probability.

We can find the upper bound of this probability by multiplying each pairwise error probability $P_2(d)$ by the corresponding number of incorrectly decoded information bits for each possible incorrect path that merges with the correct path at the Bth node, and summing over all d.

We can find the number of incorrect information bits by differentiating the transfer function $T(D, N)$ with respect to N as the exponent of N represents the number of incorrect information bits:

$$\frac{dT(D,N)}{dN} = \sum_{d=d_{\text{free}}}^{\infty} a_d f(d) D^d \qquad \text{(8-2-24)}$$

Taking the derivative with respect to N and setting $N = 1$:

$$\frac{dT(D,N)}{dN} = \sum_{d=d_{\text{free}}}^{\infty} a_d D^d \cdot \frac{d}{dN}\left(N^{f(d)}\right)$$

$$\left.\frac{dT(D,N)}{dN}\right|_{N=1} = \sum_{d=d_{\text{free}}}^{\infty} a_d f(d) D^d = \sum_{d=d_{\text{free}}}^{\infty} \beta_d D^d \qquad \text{(8-2-25)}$$

where β_d is the total information bit errors at Hamming distance d.

Now, multiplying β_d with $P_2(d)$, we get the bit error probability for $k = 1$:

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

$$P_b < \sum_{d=d_{\text{free}}}^{\infty} \beta_d Q\left(\sqrt{2\gamma_b R_c d}\right) \qquad \text{(8-2-26)}$$

Explanation

This extends the error analysis to bit error rate (BER) by considering the number of information bits in error (β_d) for each diverging path.

It refines the analysis by accounting for how many bits are wrong, not just that an error occurred.

4.Hard Decision Decoding Analysis

Again we would use the property of linearity of convolutional codes and start by finding the first error instance for an all-zero sequence:

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

But we did not consider the case where the metrics are equal, for that case the new equation would be:

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

We found the case for the first error, but now we need to find the other multiple errors that could occur. a_d is the number of errors; using (8-2-28) and (8-2-29) we get:

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

$$P_e < T(D)\big|_{D=\sqrt{4p(1-p)}} \qquad \text{(8-2-32)}$$

- We make use of the value in the exponent of N of the transfer function; this is used to determine the number of bits in error, then further differentiating $T(D, N)$ with respect to N where $N = 1$:

$$P_b < \sum_{d=d_{\text{free}}}^{\infty} \beta_d P_2(d) \qquad \text{(8-2-33)}$$

- β_d are the coefficients in the expansion of $dT(D,N)/dN$.
- The $\{\beta_d\}$ are the coefficients of the derivative of $T(D,N)$, for $N=1$. For $P_2(d)$, we can use (8-2-28) and (8-2-29).

  Finally:

$$P_b < \left.\frac{dT(D,N)}{dN}\right|_{N=1,D=\sqrt{4p(1-p)}} \tag{8-2-34}$$

- When $k>1$, the results given in (8-2-33) and (8-2-34) for P_b should be divided by k.

5.Comparison and Conclusion on Soft and Hard Decision Decoding

Soft Decision Decoding vs. Hard Decision Decoding

- **Signal Information Utilization**: Hard-decision decoding simplifies the received signal into binary values (0 or 1), losing nuanced reliability information. In contrast, soft-decision decoding preserves the signal's probability data, using metrics like Euclidean distances to make more informed decisions. This approach typically results in a 2–3 dB coding gain, enabling soft-decision decoding to maintain lower error rates at reduced signal-to-noise ratios (SNR).
- **Error Rate Performance**: Empirical studies (refer to Section 6) demonstrate that soft-decision decoding consistently achieves lower bit error rates (BER) and block error rates (BLER) across various constraint lengths, showcasing its robustness in high-noise scenarios.

Impact of Code Parameters

- **Constraint Length (K_c)**: A larger constraint length (e.g., increasing from 3 to 6) extends the code's free distance (d_{free}), improving its ability to correct errors and lowering the BER for a given SNR. However, this also increases the decoder's complexity, as the number of states (2^{K_c-1}) grows exponentially, posing computational challenges.
- **Code Rate ($R_c = k/n$)**: Reducing the code rate (e.g., from 1/2 to 1/3) adds more redundancy by increasing the output bits (n) per input bit (k). This enhances d_{free} and improves BER, but at the expense of bandwidth efficiency, requiring a trade-off based on system requirements.

In summary, soft-decision decoding offers superior error correction by utilizing signal reliability, though it demands more computational resources. Adjusting constraint length and code rate allows for optimization, balancing performance, complexity, and bandwidth efficiency for specific applications. Below are some of the graphs we derived.

6.Simulation Result

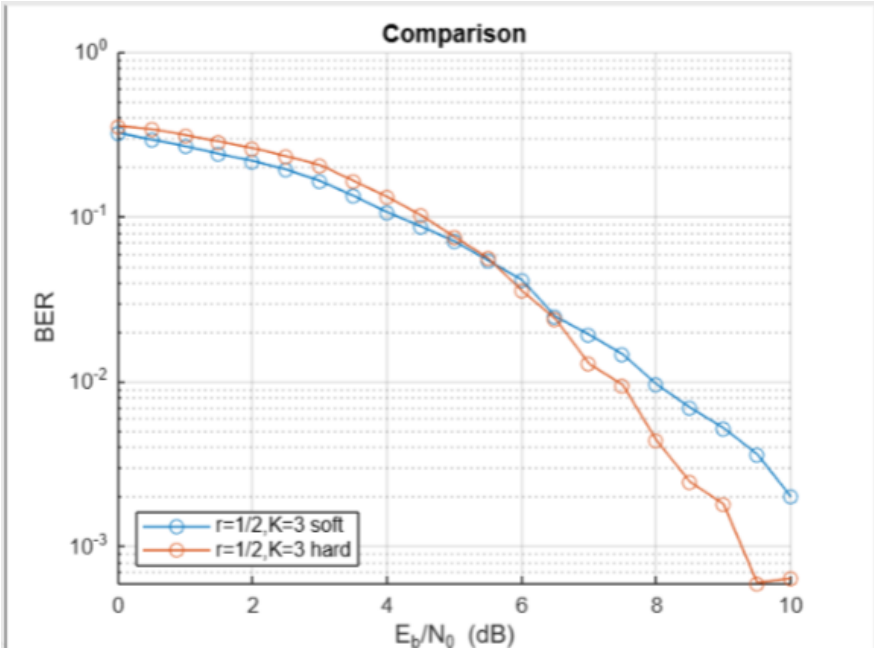

Figure 1: Comparing hard and soft decoding for rate = 1/2 and K = 3

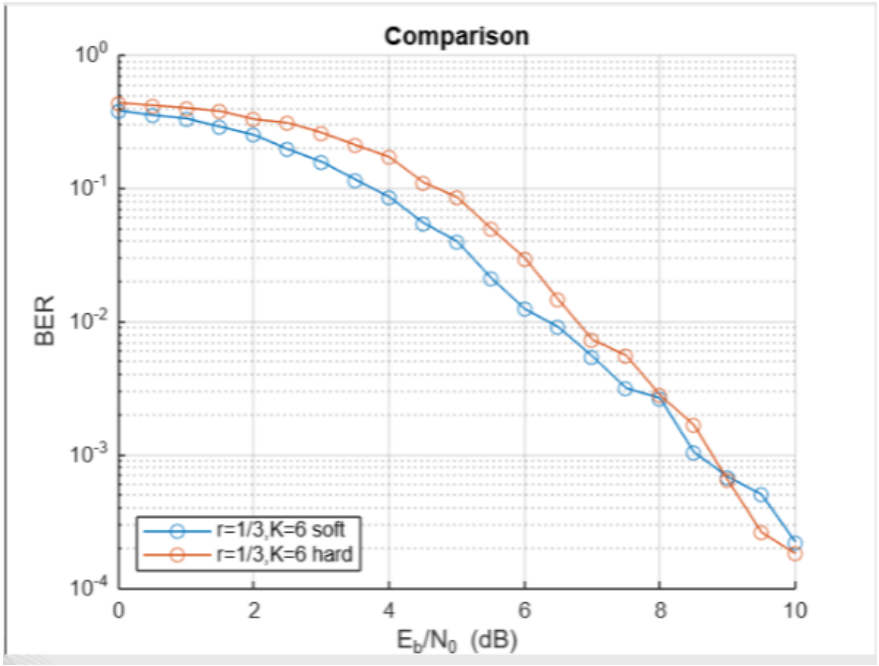

Figure 2: Comparing hard and soft decoding for rate = 1/3 and K = 6

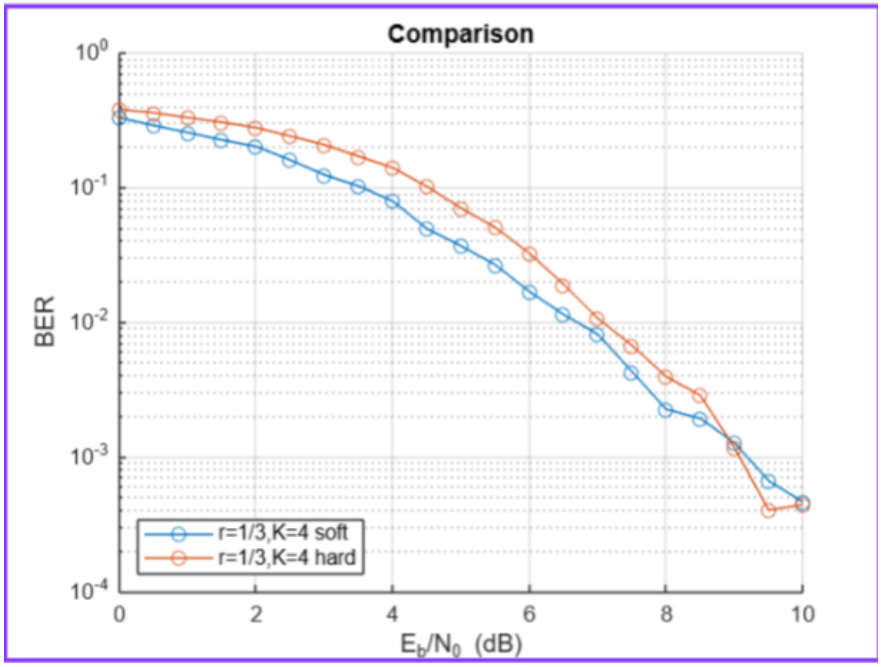

Figure 3: Comparing hard and soft decoding for rate = 1/3 and K = 4

References

1.Proakis, J. G. Digital Communications. 4th ed., McGraw-Hill, 1995

2.Hossam Labib Abdel Fattah Zayed$_V 6 I3 - IJERTV6IS030465$

3.$TeachingConvolutionalCodingusingMATLABinCommunicationSystemsCourse$

4.$Lecture8onConvolutionalCodes.MITOpenCourseWare.Availableat : https : //web.mit.edu/6.02/www/f2011/handouts/8.pdf.$

5.$Lecture9onConvolutionalCodes.MITOpenCourseWare.Availableat : https : //web.mit.edu/6.02/www/f2010/handouts/lectures/L9.pdf.$