CT216 Introduction to Communication Systems

Prof. Yash Vasavada

Convolutional Coding Analysis

By Lab-2 (Group-3 , Mentor TA : Naisheel)

(202201105) Rishi Jigar Shah
(202201110) Dhruv Gaurav Suri
(202201119) Sahilkumar Sutariya
(202201132) David Lochan Baro
(202201140) Gajjar Harsh Dharmendrabhai
(202201141) Jani Ayushi Yogeshbhai
(202201143) Vegad Rutvik Rakeshbhai
(202201144) Dhangar Dhruv Kishorbhai
(202201152) Patel Ragan Dalpatbhai
(202201153) Dodiya Dev Manojbhai

Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT)

Viterbi Algorithm:

→ In the decoding of a block code for a memoryless channel, we computed the distances (Hamming distance for hard-decision decoding and euclidean distance for soft-decision decoding) between the received code word and the 2^k possible transmitted code words. This decision rule, which requires the computation of 2^k metrics, is optimum in the sense that it results in a minimum probability of error for the binary symmetric channel with P < 1/2 and the additive white gaussian noise channel.

→ Optimum decoding of convolutional code involves a search through the trellis for the most probable sequence. Depending on whether the detector following the demodulator performs hard or soft decisions, the corresponding metric in the trellis search may be either a Hamming metric or a euclidean metric, respectively.

→ Let suppose we transmitted a message represented by "c" then it has every bit as c_{jm} where j indicates the j^{th} branch and m indicates the m^{th} bit of that branch.

$$r_{jm} = \sqrt{\varepsilon_c}\left(2C_{jm} - 1\right) + n_{jm}$$

where ε_c is transmitted signal energy and n_{jm} represents additive noise.

→ Then, we can represent the branch metric for i^{th} path as

$$\mu_j^{(i)} = log\, P\left(Y_j \middle| C_j{}^{(i)}\right)$$

where Y_j is the received branch.

→ Then, a metric of i^{th} path consisting of B branches

$$PM^{(i)} = \sum_{i=1}^{n} \mu_j^{(i)}$$

→ Above equation of path metrics calculates Hamming distance, which is an equivalent metric for hard-decision decoding.

→ Similarly, suppose the soft-decision decoding is employed and the channel adds white gaussian noise to the signal then the demodulator output is described statistically by probability density function

$$P\left(r_{jm} \middle| C_{jm}{}^{(i)}\right) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left[\frac{-\left[r_{jm} - \sqrt{\varepsilon_c}\left(2C_{jm}{}^{(i)} - 1\right)\right]^2}{2\sigma^2}\right]$$

where $\sigma^2 = N_0/2$ is variance of additive gaussian noise.

→ If we neglect the terms that are common to all branch metrics, the branch metric for j^{th} branch of the i^{th} path is expressed as

$$\mu_j^{(i)} = \sum_{i=1}^{n} r_{jm}\left(2C_{jm}{}^{(i)} - 1\right)$$

and the correlation metric for i^{th} path is

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

→ We consider two paths: one is all-zero path and other is any other possible path, which merges at a state B. Consequently, if $CM(0) > CM(1)$ at a merge node B, $CM(0)$ will continue to be larger than $CM(1)$ for any path that stems from node B. This means, $CM(1)$ can be discarded from further considerations and path corresponding to $CM(0)$ is the survivor.

→ So, as we have branch metrics and path metrics for both soft and hard decision decoding, we can use them for Viterbi Algorithm for optimization.

Soft-decision Decoding:

→ While deriving the probability of error for convolution code, the linearity property for this type of codes employed to simplify the derivation. Assume that the all-zero sequence is transmitted and we determine the probability of error in deciding the favour of another sequence. The coded binary digits for the j^{th} branch of the code, denoted as C_{jm} are assumed to be transmitted by binary-PSK and detected coherently at the demodulator. The output of demodulator which is the input to the Viterbi decoder, is r_{jm}.

→ Then, the path metrics for the soft Viterbi decoder having B branches are expressed as

$$CM^{(i)} = \sum_{j=1}^{B} \mu_j^{(i)}$$

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

→ The path metric for all-zero path$(C_{jm} = 0)$ is

$$CM^{(0)} = \sum_{j=1}^{B} \sum_{m=1}^{n} \left(\sqrt{\varepsilon_c}(2C_{jm} - 1) + n_{jm}\right)\left(2C_{jm}{}^{(i)} - 1\right)$$

$$CM^{(0)} = \sum_{j=1}^{B} \sum_{m=1}^{n} \left(-\sqrt{\varepsilon_c} + n_{jm}\right)(-1)$$

$$CM^{(0)} = Bn\sqrt{\varepsilon_c} + \sum_{j=1}^{B} \sum_{m=1}^{n} n_{jm}$$

→ As convolution code doesn't have any fixed length, so we derive its performance from the probability of error for sequence that merge with all non-zero sequence for first time at node B.

→ Let an incorrect path$(i = 1)$ merging with all-zero paths at node B differ from all-zero paths in d bits.

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

$$P_2(d) = P\left[2\sum_{j=1}^{B}\sum_{m=1}^{n} r_{jm}\left({C_{jm}}^{(1)} - {C_{jm}}^{(0)}\right) \geq 0\right]$$

→ Since, $CM(1)$ and $CM(0)$ paths differ only for d positions, except that both are same.

$$P_2(d) = P\left(\sum_{l=1}^{d} {r_l}' \geq 0\right)$$

where ${r_l}' \overset{IID}{\longrightarrow} Gaussian\ (-\sqrt{E_c}\,, N_0/2)$, the index l runs over set of d bits in which two paths differ.

→ The expectation and variance of ${r_l}'$ can be determined as

$$\begin{aligned} E\{{r_l}'\} &= E\left\{\sqrt{\varepsilon_c}(0-1) + n_{jm}\right\} \\ &= E\left\{-\sqrt{\varepsilon_c} + n_{jm}\right\} \\ &= E\left\{-\sqrt{\varepsilon_c}\right\} + E\left\{n_{jm}\right\} \\ &= -\sqrt{\varepsilon_c} \end{aligned}$$

$$\begin{aligned} var\{{r_l}'\} &= var\left\{-\sqrt{\varepsilon_c} + n_{jm}\right\} \\ &= var\left\{-\sqrt{\varepsilon_c}\right\} + var\left\{n_{jm}\right\} \\ &= \frac{N_0}{2} \end{aligned}$$

$$\therefore \sum_{l=1}^{d} {r_l}' \sim N\left(-d\sqrt{\varepsilon_c}\,, \frac{dN_0}{2}\right)$$

$$P\left(\sum_{l=1}^{d} {r_l}' \geq 0\right) = P\left(\frac{\sum_{l=1}^{d} {r_l}' + d\sqrt{\varepsilon_c}}{\sqrt{\frac{dN_0}{2}}} \geq \frac{d\sqrt{\varepsilon_c}}{\sqrt{\frac{dN_0}{2}}}\right) = P\left(Z \geq \sqrt{\frac{2\varepsilon_c d}{N_0}}\right)$$

$$\therefore\ P_2(d) = Q\left(\sqrt{2\gamma_b R_c d}\right)$$

where $\gamma_b = E_b/N_0$ is the received SNR per bit and R_c is code rate, $Q(x_0) = \frac{1}{\sqrt{\pi}}\int_{x_0}^{\infty} e^{-\frac{x^2}{2}}\,dx$

→ There are many possible paths with different distances that merge with all non-zero path at node B. Transfer function $T(D)$ provides complete description of all possible path that merge with all-zero paths at node B and their distance. On performing this, we obtain an upper-bound on the first-event error probability

$$P_e \leq \sum_{d=d_{free}}^{\infty} a_d P_2(d)$$

$$\boxed{P_e \leq \sum_{d=d_{free}}^{\infty} a_d Q\left(\sqrt{2\gamma_b R_c d}\right)} \tag{8.2.21}$$

where a_d denotes the number of paths of distance d from the all-zero path that merge with all-zero path for first time.

→ Using Chernoff bound,

$$P(x \geq a) \leq e^{-at} M_x(t)$$

$$\leq e^{-at} e^{\mu t + \frac{1}{2}\sigma^2 t^2}$$

$$\leq e^{-at + \frac{1}{2}t^2}$$

$$\Rightarrow Q(a) = P(x \geq a) \leq e^{-\frac{a^2}{2}}$$

$$\boxed{\therefore\ Q\left(\sqrt{\gamma_b R_c d}\right) \leq e^{-\frac{\gamma_b R_c d}{2}} = D^d\Big|_{n=e^{\gamma_b R_c}}} \tag{8.2.22}$$

→ From the above two equations,

$$P_e < T(D)\big|_{D=e^{\gamma_b R_c}}$$

→ Although the first-event error probability provides a measure of the performance of a convolution code, a more useful measure of performance is the bit error probability. If we multiply the pairwise error probability $P_2(d)$ by the number of incorrectly decoded information bits for the incorrect path at the node where they merge, we obtain the bit error rate for the path. The average bit probability is upper-bounded by differentiating the transfer function $T(D, N)$ with respect to N and setting $N = 1$.

$$T(D, N) = \sum_{d=d_{free}}^{\infty} a_d D^d N^{f(d)}$$

→ Differentiating both sides,

$$\left.\frac{dT(D, N)}{dx}\right|_{N=1} = \sum_{d=d_{free}}^{\infty} a_d D^d f(d)$$

$$= \sum_{d=d_{free}}^{\infty} \beta^d D^d$$

where $\beta^d = a_d f(d)$.

→ Thus, the bit error probability for $k = 1$ is upper-bounded by

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

$$\boxed{P_b < \sum_{d=d_{free}}^{\infty} \beta^d Q\left(\sqrt{2\gamma_b R_c d}\right)} \tag{8.2.26}$$

→ If the Q function is upper-bounded by an exponential as indicated in (8.2.22) then (8.2.26) can be expressed in the simple form

$$P_b < \sum_{d=d_{free}}^{\infty} \beta^d D^d \Big|_{D=e^{\gamma_b R_c}}$$

$$< \frac{dT(D,N)}{dN}\Bigg|_{N=1,D=e^{\gamma_b R_c}}$$

Hard-decision Decoding:

→ Let suppose the path being compared to all-zeros path at some node B has distance d from all-zeros path. We consider two cases,

- When d is odd: If number of errors in the received sequence is less than $\frac{1}{2}(d+1)$, then it is the correct path. Otherwise, incorrect path.

  Problem of selecting incorrect path:

$$P_2(d) = \sum_{k=\frac{(d+1)}{2}}^{d} \binom{d}{k} p^k (1-p)^{d-k} \tag{8.2.28}$$

- When d is even: If the number of errors is less than $\frac{d}{2}$, then it is the correct path. Otherwise, incorrect path.

  Problem of selecting incorrect path:

$$P_2(d) = \sum_{k=\frac{d}{2}+1}^{d} \binom{d}{k} p^k (1-p)^{d-k} + \frac{1}{2}\binom{d}{\frac{1}{2}d} p^{d/2}(1-p)^{d/2} \tag{8.2.29}$$

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

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

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

$$= 2^d \, p^{\frac{d}{2}} \, (1-p)^{\frac{d}{2}}$$

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

→ There are many possible paths with different distances merging with all-zero path at a given node. Therefore, there is no exact expression for the first-event error probability. However, we can upper-bound it by

$$Probability\ of\ error\ (P_e) < \sum_{d=d_{free}}^{\infty} a_d P_2(d) \tag{8.2.30}$$

where a_d = the number of paths corresponding to set of distances $\{d\}$.

→ We can replace $P_2(d)$ by its connection with transfer function:

$$\boxed{P_2(d) < [4p(1-p)]^{d/2}} \tag{8.2.31}$$

→ Therefore, putting value of $P_2(d)$ in $(8.2.30)$, we have

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

$$\boxed{P_e < T(D)|_{D=\sqrt{4p(1-p)}}} \tag{8.2.32}$$

→ Upper-bound on the bit error probability gives

$$P_b < \sum_{d=d_{free}}^{\infty} \beta_d \, P_2(d) \tag{8.2.33}$$

where β_d = the coefficient in the expansion of the derivative of $T(D, N)$ evaluated as $N = 1$

$$\boxed{\therefore\ P_b < \frac{dT(D,N)}{dN}\bigg|_{N=1, D=\sqrt{4p(1-p)}}} \tag{8.2.34}$$

→ Above equations $(8.2.23)$ and $(8.2.34)$ are for $k = 1$. If $k > 1$, then simply divide them by k.

Comparison between Hard-decision and Soft-decision Decoding:

→ The simulation results indicate that the Soft-decision Decoding is more efficient than the Hard-decision Decoding. The reason for this is that Soft-decision Decoding determines the level of informational value of each segment.

→ While Hard-decision Decoding entails straightforward decision-making, the bit's informativeness is not taken into account.

→ In the same way, the bit error rate drops with increasing constraint length. The redundancy rises as the constraint length does. This helps in enhancing the code's overall performance in noisy environments.