# SYSTEM ARCHTECTURE LAB on 2015/Jan/10th - Convolutional Coder and Viterbi Decoder -

(1) 以下のtransfer function matrixに対する ENCODERおよびVITERBIデコードを実現する

$$G_a(x) = \begin{bmatrix} 1+x^2 & 1+x+x^2 \end{bmatrix}$$

対応する回路図は

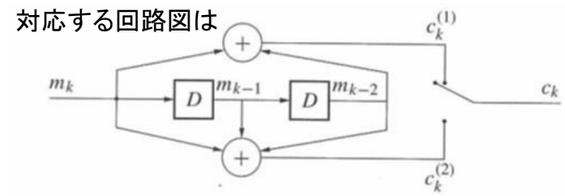
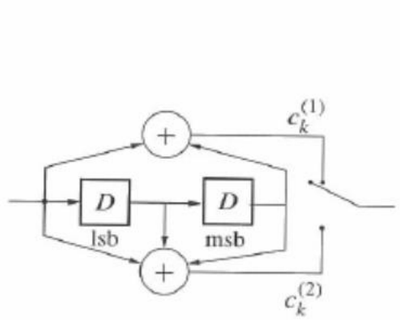
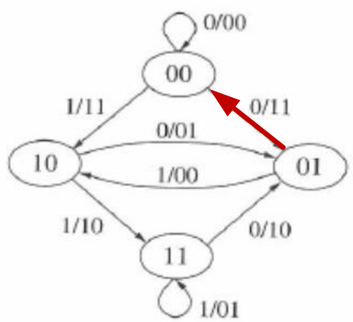



Figure 12.1: A rate $R = 1/2$ convolutional encoder.

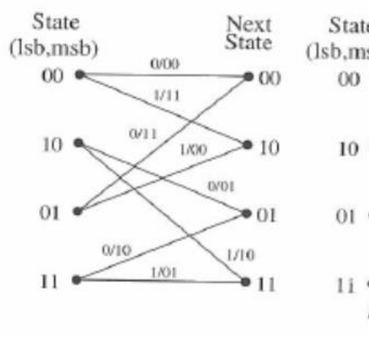
動作説明



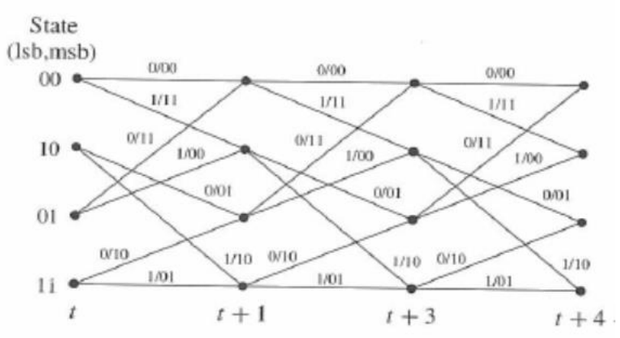

(a) Encoder.




(b) State Diagram.




(c) One stage of trellis.




(d) Three stages of trellis.

Figure 12.5: Encoder, state diagram, and trellis for $G(x) = [1 + x^2, 1 + x + x^2]$.

例題1 **Example 12.13** Consider the encoder

$$G(x) = \begin{bmatrix} x^2+1 & x^2+x+1 \end{bmatrix}$$

of Example 12.1, whose realization and trellis diagram are shown in Figure 12.5, passing the data through a BSC. When the data sequence

$$\begin{aligned} \mathbf{m} &= [1, 1, 0, 0, 1, 0, 1, 0, \ldots] \\ &= [m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7, \ldots] \end{aligned}$$

is applied to the encoder, the coded output bit sequence is

$$\begin{aligned} \mathbf{c} &= [11, 10, 10, 11, 11, 01, 00, 01, \ldots] \\ &= [\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4, \mathbf{c}_5, \mathbf{c}_6, \mathbf{c}_7, \ldots]. \end{aligned}$$

For the BSC, we take the mapped data the same as the encoder output $\mathbf{a}_t = \mathbf{c}_t$. The output sequence and corresponding states of the encoder are shown here, where $\Psi_0 = 0$ is the initial state.

| $t$ | Input $m_k$ | Output $\mathbf{c}_t$ | State $\Psi_{t+1}$ |
|---|---|---|---|
| 0 | 1 | 11 | 1 |
| 1 | 1 | 10 | 3 |
| 2 | 0 | 10 | 2 |
| 3 | 0 | 11 | 0 |
| 4 | 1 | 11 | 1 |
| 5 | 0 | 01 | 2 |
| 6 | 1 | 00 | 1 |
| 7 | 0 | 01 | 2 |

The sequence of states through the trellis for this encoder is shown in Figure 12.12; the solid line shows the state sequence for this sequence of outputs. The coded output sequence passes through a

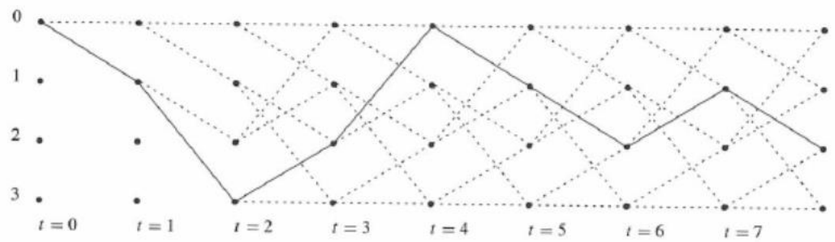



Figure 12.12: Path through trellis corresponding to true sequence.

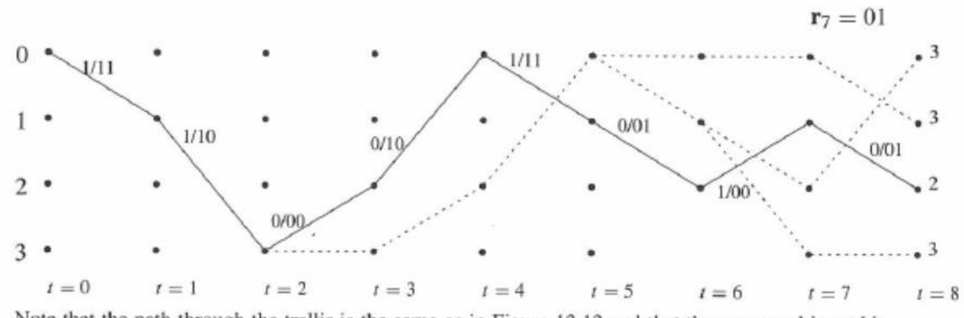
channel, producing the received sequence

2ビットエラーを発生

$$\mathbf{r} = [11\,10\,\underline{0}0\,1\underline{0}\,11\,01\,00\,01\ \ldots] = [\mathbf{r}_0, \mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_4, \mathbf{r}_5, \mathbf{r}_6, \mathbf{r}_7, \ldots].$$

The two underlined bits are flipped by noise in the channel.

The algorithm proceeds as follows:




Note that the path through the trellis is the same as in Figure 12.12 and that the recovered input bit sequence is the same as the original bit sequence. Thus, out of this sequence of 16 bits, two bit errors have been corrected. □

**Example 12.15** The table

上記グラフの実装のためのデータTABLE

| $t$: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| State | Previous State/Input | | | | | | | |
| 0: | 0/0 | 0/0 | 0/0 | **2/0** | 2/0 | 0/0 | 0/0 | 2/0 |
| 1: | **0/1** | 0/1 | **2/1** | 2/1 | **0/1** | 0/1 | **2/1** | 0/1 |
| 2: | - | 1/0 | **3/0** | 3/0 | 1/0 | **1/0** | 1/0 | **1/0** |
| 3: | - | **1/1** | 3/1 | 2/1 | 1/1 | 3/1 | 1/1 | 3/1 |

shows the previous state traceback table which would be built up by the decoding of Example 12.13. For example, at time $t = 8$, the predecessor of state 0 is 2, the predecessor of state 1 is 0, and so forth. Starting from state 2 (having the lowest path cost), the sequence of states can be read off in reverse order from this table (the bold entries):

$$2 \to 1 \to 2 \to 1 \to 0 \to 2 \to 3 \to 1 \to 0.$$

Thus the first state transition is from state 0 to state 1 and the input at that time is a 1.

The inputs for the entire sequence can also be read off, starting at the right, 11001010. □

(2) 前ページ(1)を以下のようにBPSK変調し、1000ビット情報の送信、受信に拡張する
ただし、1000ビットの最後の2ビットは"00"とし、ENCODERがSTATE=00に最後もどるようにする

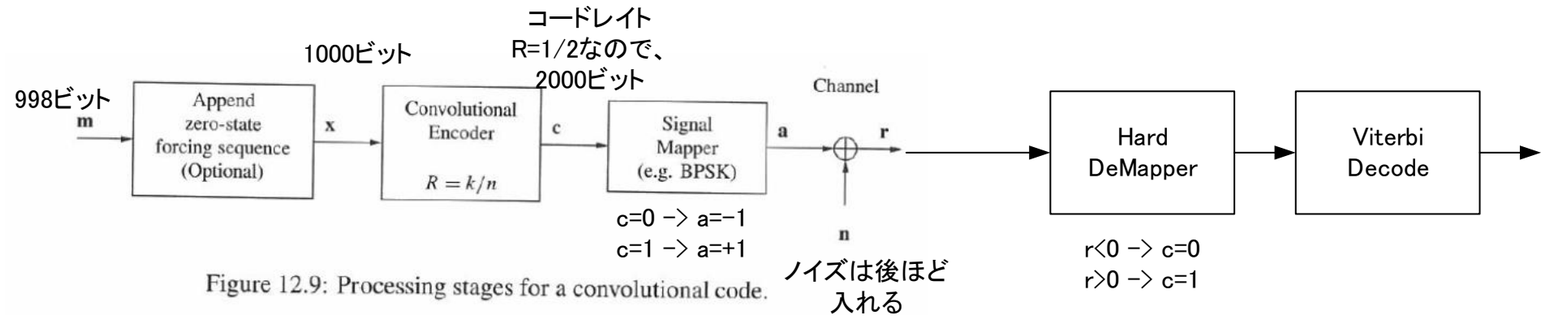
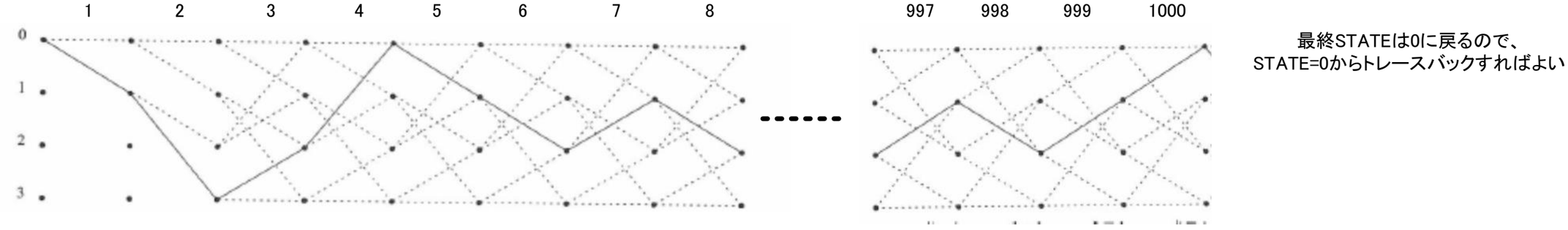


Figure 12.9: Processing stages for a convolutional code.




最終STATEは0に戻るので、
STATE=0からトレースバックすればよい

(3) 上記(2)を以下のようにSOFT DEMAPPERに変更する

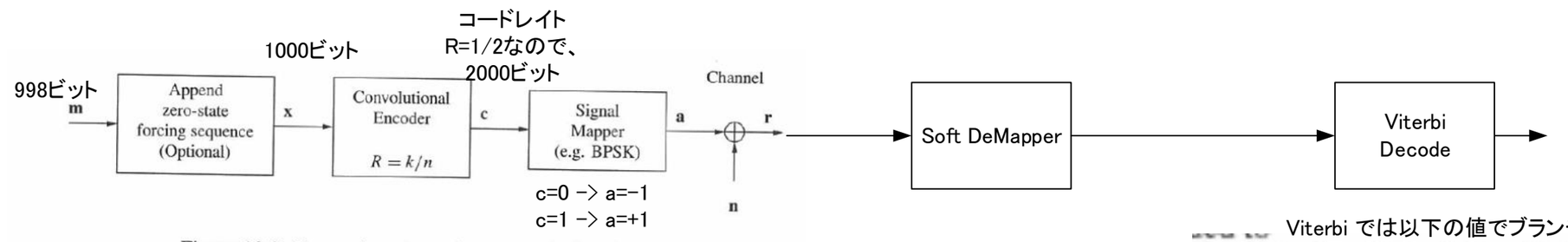


Figure 12.9: Processing stages for a convolutional code.

$$q_t = Q[r_t] = \begin{cases} 00 & \text{if } r_t \in \mathcal{R}_{00} = (-\infty, -1] \\ 01 & \text{if } r_t \in \mathcal{R}_{01} = (-1, 0] \\ 10 & \text{if } r_t \in \mathcal{R}_{10} = (0, 1] \\ 11 & \text{if } r_t \in \mathcal{R}_{11} = (1, \infty). \end{cases}$$

Viterbi では以下の値でブランチメトリックを計算する

| $\lfloor a(-\log(P(q_t \mid a) - b) \rfloor$ | $q_t =$ | 00 | 01 | 10 | 11 |
|---|---|---|---|---|---|
| $a = 1$ C=1 | | **3** | 2 | 1 | 0 |
| $a = -1$ C=0 | | 0 | 1 | 2 | **3** |

(4) 上記(3)のSOFT DEMAPPERを3ビット化して実装せよ

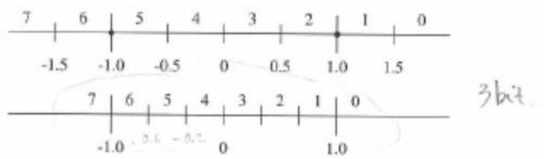
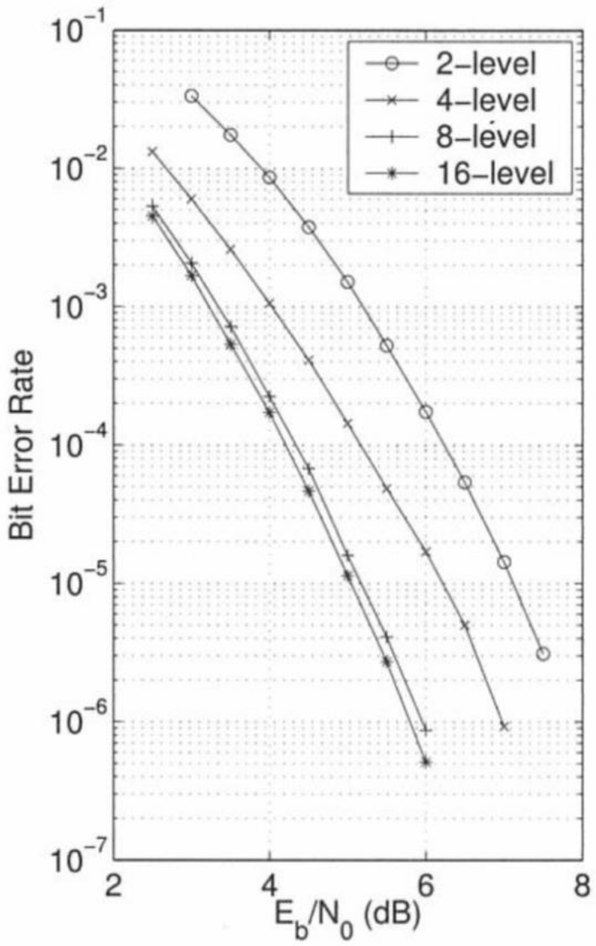


Figure 12.15: Quantization thresholds for 4- and 8-level quantization.

Table 12.1: Quantized Branch Metrics Using Linear Quantization

| Signal Amplitude | Quantization Level 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| | Branch Metric $\mu$ | | | | | | | |
| −1 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

(5) 前ページ(2), (3), (4)にノイズを入れて以下のようなグラフを作成せよ

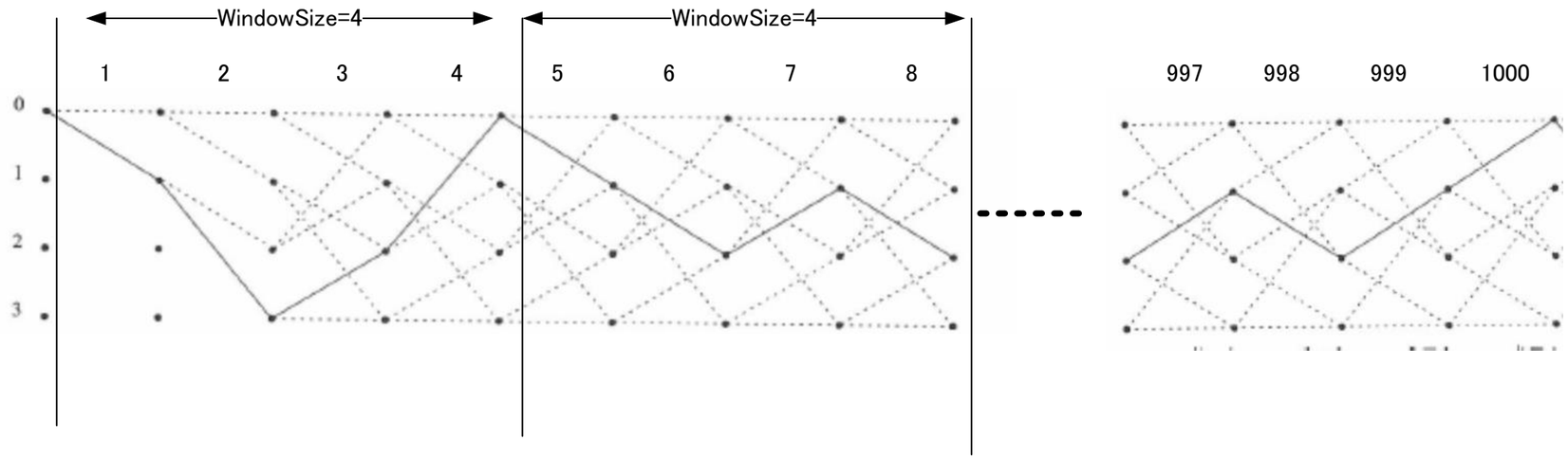



(a) 2-, 4-, 8- and 16-level quantization, decoding window 32.

X軸は、信号(−1, +1)の平均パワー＝1　と　ランダムノイズパワーの平均値の比のdB表示

(6) 前ページ(2), (3), (4)をWINDOW処理でデコードせよWindowSize=5, 10, 20, 40を比較せよ

以下の例では8まで進んだ時に、STATE=0からトレースバックして、4までのWINDOWのデータをデコードする。
そして、12まで進み、また、STATE=0からトレースバックして、5から8までのデコードを行う、
上記を繰り返す。
最後は、1000でSTATE=0になることが約束されているので、それを利用する。




最終レポート課題、
中間レポートを書いてもらった、以下(12.49)式のコンボリューショナルデコーダに対して、
上記、(1)から(6)の処理プログラムを作成し、中間レポートで書いてもらったデコーダの説明、
今回の処理コード、結果を含むレポートを作成してください。

最終レポート提出日：　2月8日　和田まで電子メールで送付

## 12.11 Exercises

12.1 For the $R = 1/2$ convolutional encoder with

$$G(x) = [1 + x^2 + x^3 \quad 1 + x + x^3] \tag{12.49}$$