Parity Check Trellis Representation of Linear Block Codes

Keith M. Chugg

March 26, 2025

An (n, k) linear, binary block code with a given parity check matrix

$$\mathbf{H} = \left[\begin{array}{cccc} \mathbf{h}_0 & \mathbf{h}_1 & \cdots & \mathbf{h}_{n-1} \end{array} \right] \tag{1}$$

where $\mathbf{h}_j$ is the $((n-k) \times 1)$ vector making up the j^{th} column for $j = 0, 1, \ldots (n-1)$. Define the *partial syndrome* of a vector $\mathbf{y}$ as

$$\mathbf{s}_j(\mathbf{y}_0^{j-1}) = \sum_{m=0}^{j-1} y_m \mathbf{h}_m \tag{2}$$

Then, $\mathbf{s}_n(\mathbf{y}_0^{n-1})$ is the (full) syndrome associated with the binary vector $\mathbf{y}$.

The partial syndrome takes on 2^{n-1} distinct values since $\mathbf{H}$ is rank $n-k$. Furthermore, given $\mathbf{s}_j(\mathbf{y}_0^{j-1})$ and y_j, the value of $\mathbf{s}_{j+1}(\mathbf{y}_0^{j})$ is determined. This update is initialized with $\mathbf{s}_0(\cdot) = \mathbf{0}$. Thus, this is a finite machine (FSM) model to describe the constraint imposed by the code[1] with state at time j defined as $s_j \equiv \mathbf{s}_j(\mathbf{y}_0^{j-1})$ (*e.g.,* one can establish a labeling convention on the partial syndrome vector values to integers in $\{0, 1, \ldots N_s\}$ where $N - s \leq (n-k-1)$).

One immediate consequence of this FSM model is that since $\mathbf{Hc} = \mathbf{0}$ for all codewords $\mathbf{c}$, all codewords correspond to paths through a trellis representing this FSM that terminate into the final state of zero (*i.e.,* zero syndrome). Consider the example of a $(8, 6)$ code with parity check matrix

$$\mathbf{H} = \left[\begin{array}{cccccccc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array} \right] \tag{3}$$

The parity check trellis for this code is shown in Fig. 1 where only paths corresponding to valid codeword are shown (*i.e.,* all paths terminate into the zero state). Transitions from s_j to s_{j+1} corresponding to $c_j = 0$ are dashed and those corresponding to $c_j = 1$ are solid.

With a parity check trellis representation for a code one can decode the code via the Viterbi algorithm or forward-backward algorithm (*i.e.,* HIHO, SIHO, or SISO decoding). To accomplish soft-in decoding, the transition metrics are taken to be $\overline{\text{MI}}[c_j]$ for transitions corresponding to c_j. The only difference between this and decoding of a convolutional code is the time-varying nature of the trellis. Note that if non-uniform soft-in information is available on the input bits $\{b_i\}$, then these can be easily accounted for if the $\mathbf{H}$ matrix is in systematic form. Otherwise, updating soft information on the information bits using SISO based on the parity check trellis is not straightforward. To perform ML codeword decoding on the BSC channel, take the transition metrics to be the Hamming difference between the received binary symbol y_j and the corresponding value of c_j.

[1]Here, y_j can be viewed as the input with no outputs.

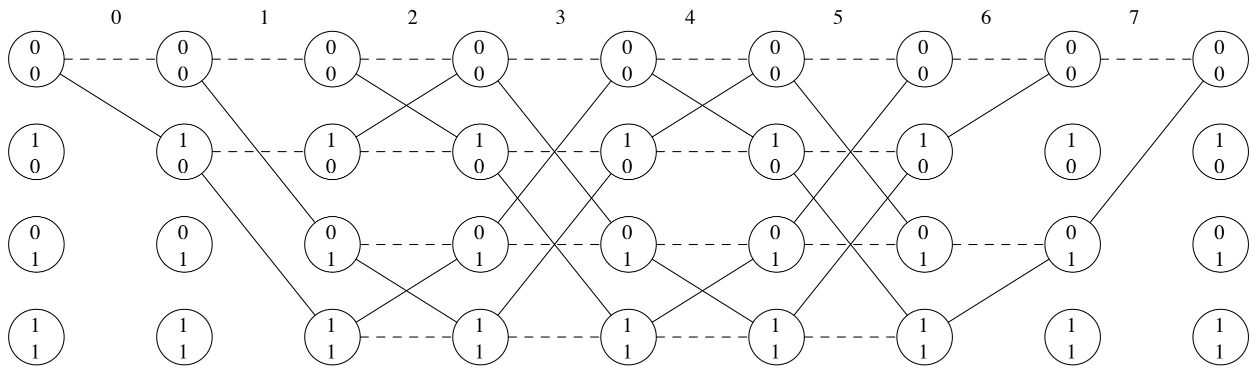

Figure 1: The parity check trellis for the $(8, 6)$ code. States are labeled by their partial syndrome values.

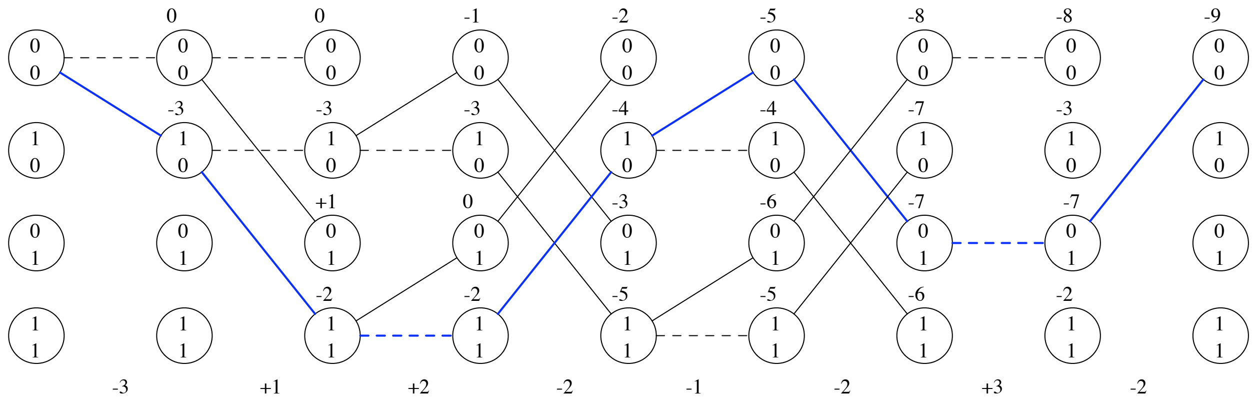

Figure 2: Decoding the example code using the Viterbi algorithm on the parity check trellis.

As an example, consider SIHO decoding of this example code with uniform, independent information bits. Fig. 2 shows the result of running the Viterbi algorithm to decode the code with the normalized channel metrics $\overline{\mathrm{MI}}[c_j]$ shown below the trellis. Note that the codeword decision is $\hat{\mathbf{c}}^{\mathrm{t}} = (11011101)$. SISO decoding can be accomplished using the forward-backward algorithm in a similar manner.

The coset leaders can also be found using a slight variation of the parity check trellis. Specifically, consider all syndromes instead only paths terminating into the zero syndrome. This is shown in Fig. 3 for the example given in (3). The coset leaders can be found using the forward state recursion with transition metrics being the Hamming weight of associated with the trellis transition. Note that this will yield survivors for each of the 2^{n-k} states at the end of the trellis. These survivors are the minimum weight sequences $\mathbf{e}$ such that $\mathbf{He}$ is the syndrome given by the final state – *i.e.,* these final survivors are the coset leaders.

This process is illustrated in Fig. 4 where the VA is run on the trellis of Fig. 3 with Hamming weight metrics. Ties in the ACS process have been shown by leaving both transitions. Note that

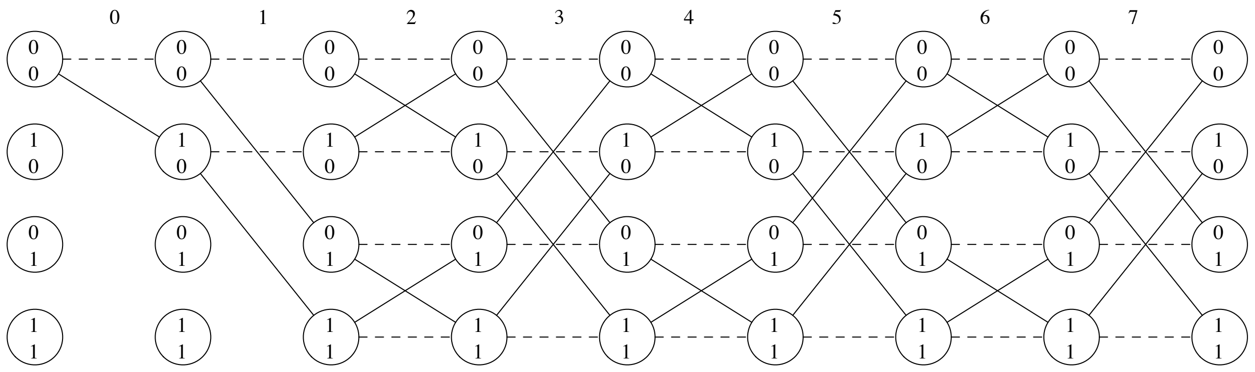

Figure 3: The unterminated parity check trellis for the $(8, 6)$ code. This can be used to find the coset leaders.

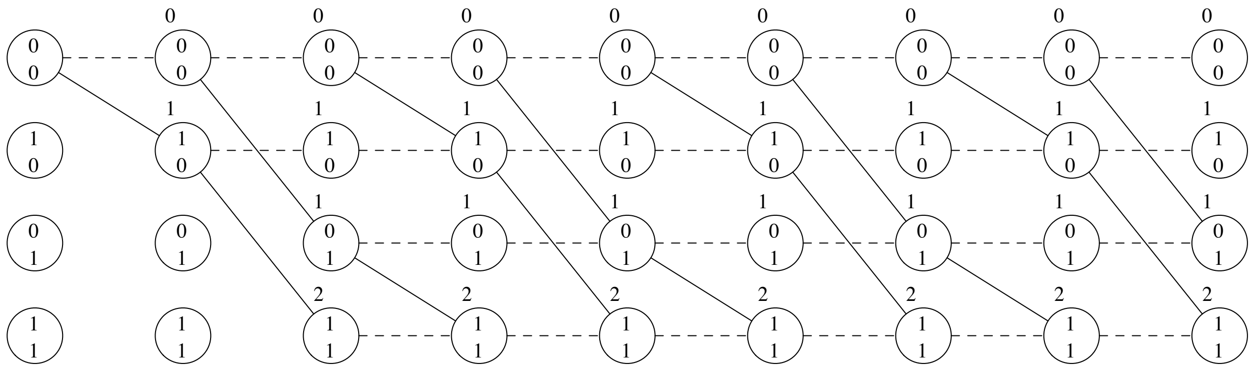

Figure 4: The results of running the Viterbi algorithm on the unterminated parity check trellis to determine the coset leaders.

there are many ties implying that the coset leaders are not unique. One choice of coset leaders is

$$\mathbf{s} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \Longrightarrow \mathbf{l}^{\mathrm{t}}(\mathbf{s}) = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 0) \tag{4}$$

$$\mathbf{s} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \Longrightarrow \mathbf{l}^{\mathrm{t}}(\mathbf{s}) = (1\ 0\ 0\ 0\ 0\ 0\ 0\ 0) \tag{5}$$

$$\mathbf{s} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \Longrightarrow \mathbf{l}^{\mathrm{t}}(\mathbf{s}) = (0\ 1\ 0\ 0\ 0\ 0\ 0\ 0) \tag{6}$$

$$\mathbf{s} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \Longrightarrow \mathbf{l}^{\mathrm{t}}(\mathbf{s}) = (1\ 1\ 0\ 0\ 0\ 0\ 0\ 0) \tag{7}$$

Notice that this implies that code has a minimum distance smaller than 3.

Using a similar idea, one can find the minimum distance of the code from its parity check trellis. Using the same metrics based Hamming weight and the terminated trellis in Fig. 1, the forward-backward algorithm can be run with a modified completion step. The modification is to exclude the zero-state to zero-state transition in the completion step. This will eliminate the all zero path. One can find the MSM of paths consistent with the other transitions at this trellis stage. Specifically, compute

$$M_j = \min_{t_j=(s_j,c_j,s_{j+1})\neq(0,\cdot,0)} \{\mathrm{F}_{j-1}[s_j] + \mathrm{w}_H(c_j) + \mathrm{B}_{j+1}[s_{j+1}]\} \tag{8}$$

for each value of j and the minimum of these values over all j is the minimum nonzero codeword weight. Check this method to verify that the minimum distance of this code is 2.