Introduction to Binary Convolutional Codes (II)

Termination

1. The *effective code rate*, $R_{\text{effective}}$, is defined as the average number of input bits carried by an output bit.
2. In practice, the input sequences are with finite length.
3. In order to terminate a convolutional code, some bits are appended onto the information sequence such that the shift registers return to the zero.
4. Each of the k input sequences of length L bits is padded with m zeros, and these k input sequences jointly induce $n(L+m)$ output bits.
5. The effective rate of the terminated convolutional code is now

$$R_{\text{effective}} = \frac{kL}{n(L+m)} = R\frac{L}{L+m}$$

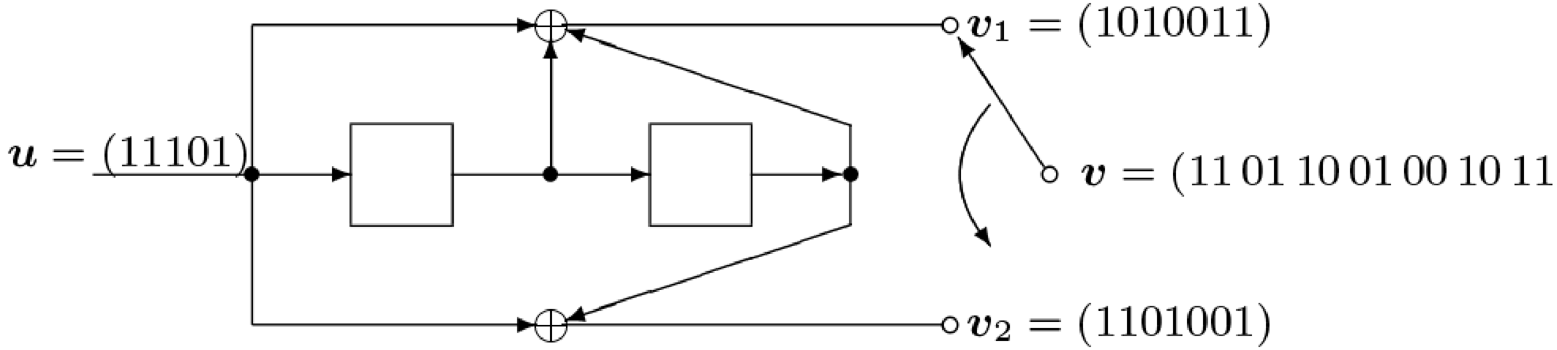
6. When L is large, $R_{\text{effective}} \approx R$.
7. All examples presented are terminated convolutional codes.

Truncation

1. The second option to terminate a convolutional code is to stop for $t > L$ no matter what contents of shift registers have. The effective code rate is still R.
2. The generator matrix is clipped after the Lth column:

$$\mathbf{G}^c_{[L]} = \begin{bmatrix} \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & & & \\ & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & & \\ & & \ddots & & \ddots & & \\ & & & \mathbf{G}_0 & & & \mathbf{G}_m \\ & & & & \ddots & & \vdots \\ & & & & & \mathbf{G}_0 & \mathbf{G}_1 \\ & & & & & & \mathbf{G}_0 \end{bmatrix}.$$

★ Example 9: Generator matrix of the truncation binary (2, 1, 2) convolutional code



$$\mathbf{v} = \mathbf{u} \cdot \mathbf{G}^{c}_{[5]} = (1, 1, 1, 0, 1) \cdot \begin{bmatrix} 11 & 10 & 11 & & \\ & 11 & 10 & 11 & \\ & & 11 & 10 & 11 \\ & & & 11 & 10 \\ & & & & 11 \end{bmatrix} = (11, 01, 10, 01, 00)$$

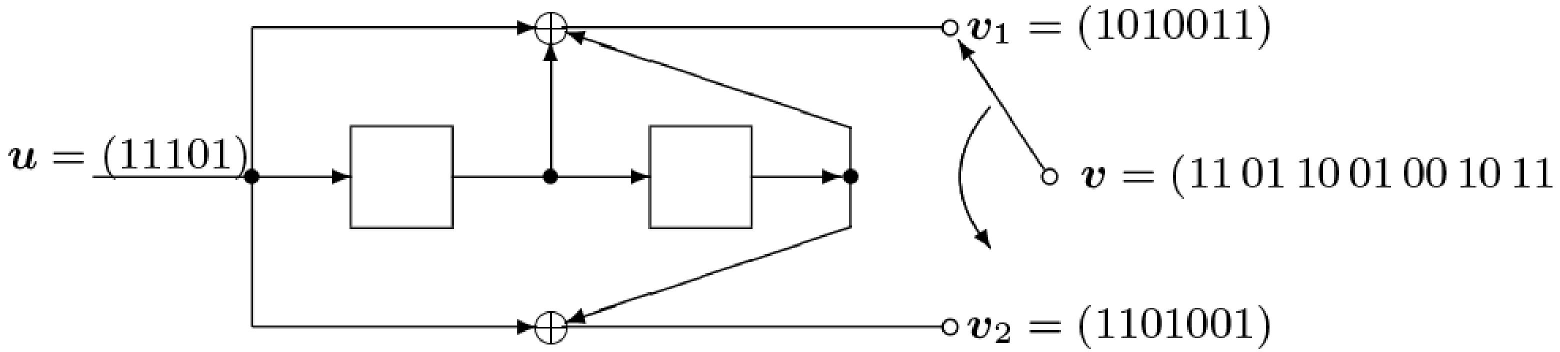
Tail Biting

1. The drawback of truncation method is that the last few blocks of information sequences are less protected.
2. Tail biting is to start the convolutional encoder in the same contents of all shift registers (state) where it will stop after the input of L information blocks.
3. Equal protection of all information bits of the entire information sequences is possible.
4. The input zeros for termination is not required. The effective rate of the code can be increased to R.
5. The generator matrix has to be clipped after the Lth column, and manipulated

as follows:

$$
\tilde{\mathbf{G}}^{c}_{[L]} = \begin{bmatrix}
\mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & & & \\
 & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & & \\
 & & \ddots & & \ddots & & \\
 & & & \mathbf{G}_0 & & & \mathbf{G}_m \\
\mathbf{G}_m & & & & \ddots & & \vdots \\
\vdots & \ddots & & & & \mathbf{G}_0 & \mathbf{G}_1 \\
\mathbf{G}_1 & \cdots & \mathbf{G}_m & & & & \mathbf{G}_0
\end{bmatrix}.
$$

★ Example 10: Generator matrix of the tail-biting binary (2, 1, 2) convolutional code



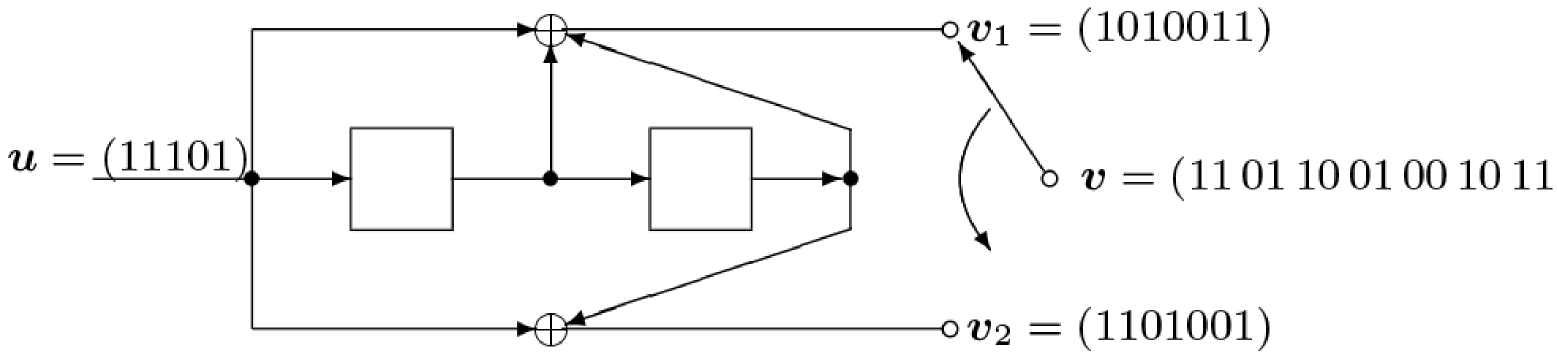
$$\mathbf{v} = \mathbf{u} \cdot \mathbf{G}^{c}_{[5]} = (1,1,1,0,1) \cdot \begin{bmatrix} 11 & 10 & 11 & & \\ & 11 & 10 & 11 & \\ & & 11 & 10 & 11 \\ 11 & & & 11 & 10 \\ 10 & 11 & & & 11 \end{bmatrix} = (01,10,10,01,00)$$

State Diagram

1. A convolutional encoder can be treated as a finite state machine.
2. The contents of the shift registers represent the states. The output of a code block $\mathbf{v}_t$ at time t depends on the current state σ_t and the information block $\mathbf{u}_t$.
3. Each change of state $\sigma_t \Rightarrow \sigma_{t+1}$ is associated with the input of an information block and the output of a code block.
4. The *state diagram* is obtained by drawing a graph. In this graph, nodes are possible states and the state transitions are labelled with the appropriate inputs and outputs ($\mathbf{u}_t/\mathbf{v}_t$). In this course we only consider the convolutional encoder with state diagrams that do not have parallel transitions.
5. The state of the encoder can be expressed as k-tuple of the memory values: $\sigma_t = (u_{1,t-1}, \cdots, u_{1,t-K_1}, u_{2,t-1}, \cdots, u_{2,t-K_2}, \cdots, u_{k,t-1}, \cdots, u_{k,t-K_k})$.
6. The *state sequence* is defined as

$$\mathbf{S} = (\sigma_0, \sigma_1, \cdots, \sigma_t, \cdots).$$

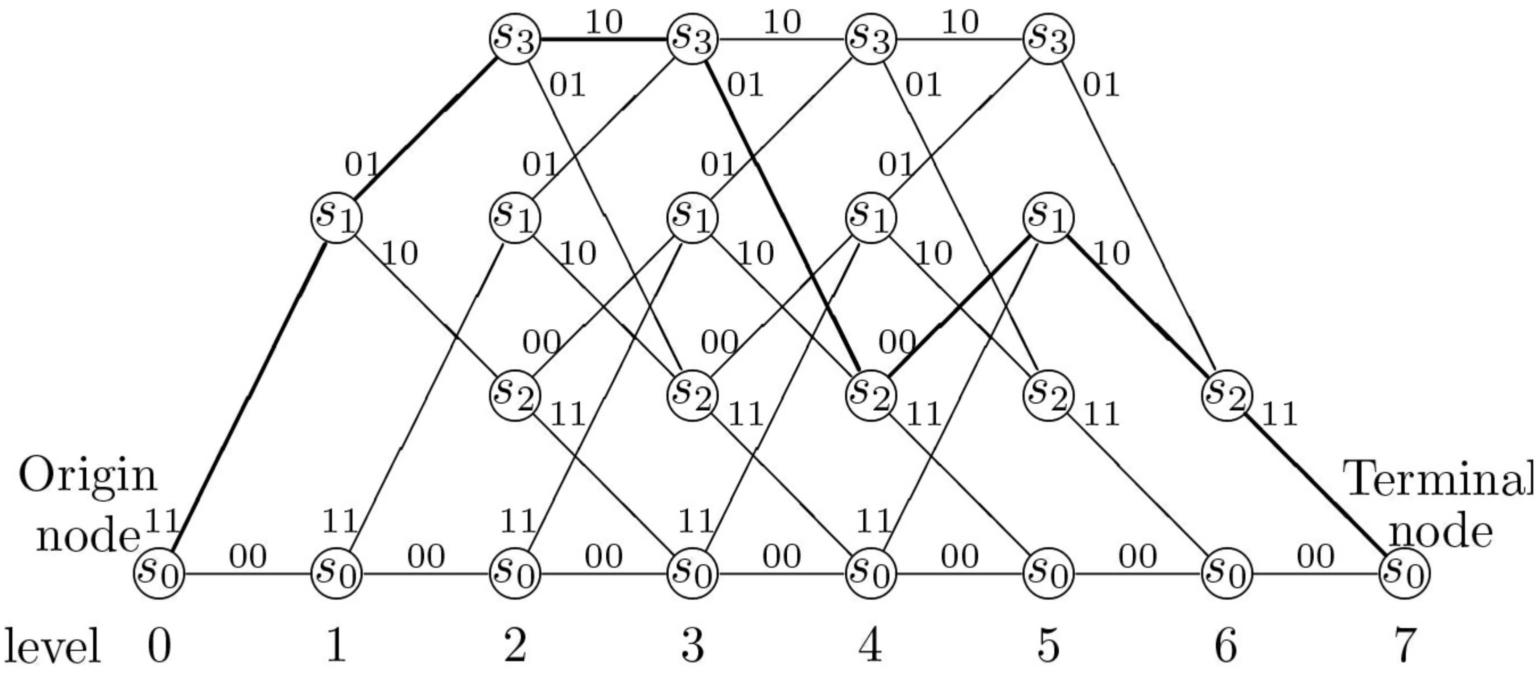
★ Example 11: State diagram of the binary (2, 1, 2) convolutional code



Trellis Diagram

1. We consider a k/n, overall constraint length K convolutional code.
2. The trellis and the state diagrams each have 2^K possible states.
3. There are 2^k branches entering each state and 2^k branches leaving each state.

★ Example 12: State diagram of the binary (2, 1, 2) convolutional code



Free Distance

1. The most important distance measure for convolutional codes is the minimum *free distance*.

2. [Def] The minimum free distance of a convolutional code is defined as

$$d_{free} \equiv \min_{\mathbf{u}',\mathbf{u}''} \{d_H(\mathbf{v}', \mathbf{v}'') : \mathbf{u}' \neq \mathbf{u}''\}.$$

   where $d_H()$ denotes the Hamming distance, $\mathbf{v}'$ and $\mathbf{v}''$ are the codewords corresponds to information bits $\mathbf{u}'$ and $\mathbf{u}''$, respectively.

3. Since convolutional code is a linear code,

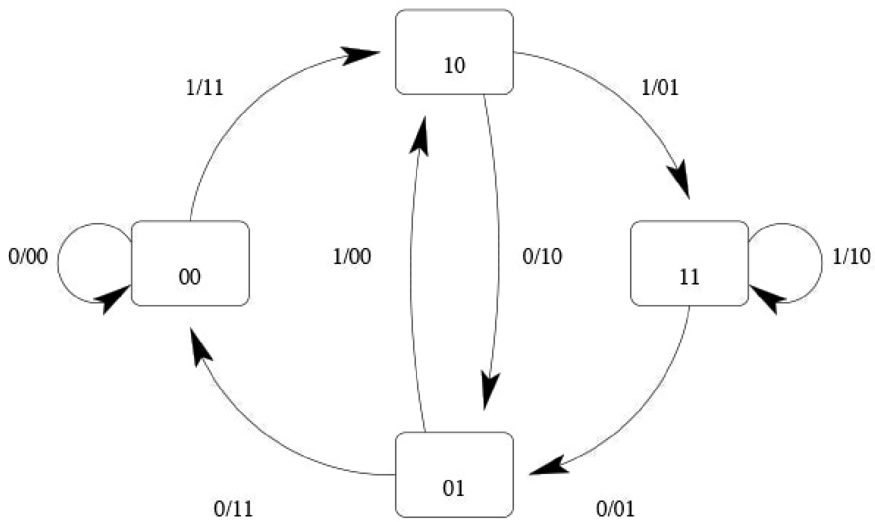
$$d_{free} = \min_{\mathbf{u}',\mathbf{u}''} \{w(\mathbf{v}' + \mathbf{v}'') : \mathbf{u}' \neq \mathbf{u}''\} = \min_{\mathbf{u}} \{w(\mathbf{v}) : \mathbf{u} \neq \mathbf{0}\}$$

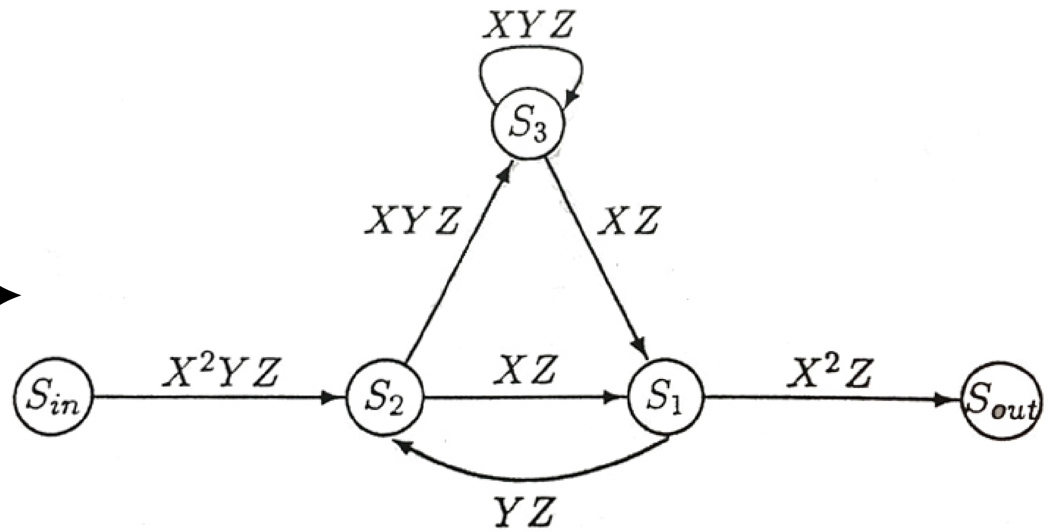
4. Hence, d_{free} is the minimum-weight codeword produced by any finite-length non-zero information sequence.

5. Also, it is the minimum weight of all finite-length paths in the state diagram that diverge from and merge with the all-zero state S_0.

Path Enumerators

1. The state diagram can be modified to provide a complete description of the Hamming weights of all nonzero code words.
2. State S_0 is split into an input state and a output state, the self-loop around state S_0 is deleted, and forms a *augmented state diagram*.
3. Each path connecting the input state to the output state represents a nonzero code word that diverge from and remerge with state S_0 exactly once.
4. The path gain is the product of the branch gains along a path.





Augmented state diagram

1. For the path gains, X^i denotes the weight of the coded bits is i, Y^i denotes the weight of the information bits is i, and Z^i denotes the number of branch is i.
2. Define A_i as the number of paths of weight i which depart from the all-zero path at a node and then remerge for the first time at a later node.
3. The set of equations describing the state transitions in the *augmented state*

diagram are

$$S_2 = YZS_1 + X^2YZS_{in}....(A)$$
$$S_1 = XZS_2 + XZS_3....(B)$$
$$S_3 = XYZS_2 + XYZS_3...(C)$$
$$S_{out} = X^2ZS_1...(D)$$

4. From (C), we get

$$S_3 = \frac{XYZS_2}{1 - XYZ}$$

5. Substitute to (B), we have

$$S_1 = \frac{XZS_2}{1 - XYZ}$$

6. Substitute to (A) and (D), we have

$$S_2 = \frac{(1 - XYZ)X^2YZS_{in}}{1 - XYZ - XYZ^2}$$
$$S_{out} = \frac{X^3Z^2S_2}{1 - XYZ}$$

7. The complete path enumerator is defined as

$$T(X,Y,Z) \equiv \frac{S_{out}}{S_{in}} = \frac{X^5 Y Z^3}{1 - XYZ(1+Z)}$$
$$= X^5 Y Z^3 + X^6 Y^2 (Z^4 + Z^5) + X^7 Y^3 (Z^5 + 2Z^6 + Z^7)$$
$$+ X^8 Y^4 (Z^6 + 3Z^7 + 3Z^8 + Z^9) + \cdots$$

8. If we set $Z = 1$,

$$T(X,Y) = X^5 Y + 2X^6 Y^2 + 4X^7 Y^3 + 8X^8 Y^4 + \cdots$$

9. If we further set $Y = 1$, the weight distribution A_i cam be obtained from

$$T(X) = X^5 + 2X^6 + 4X^7 + 8X^8 + \cdots$$

where $A_5 = 1$, $A_6 = 2$, $A_7 = 4$ and so on.

Masons Gain Formula

1. The path enumerator of a code can be determined by applying Masons gain formula to compute its generating function

$$T(X) = \frac{\sum_k F_k \Delta_k}{\Delta}$$

where

$\Delta = 1 - \sum_i C_i + \sum_{i,j} C_i C_j - \sum_{i,j,l} C_i C_j C_l + \cdots$

$\sum_i C_i$: Sum of loop gains.

$\sum_{i,j} C_i C_j$: Sum of the products of the loop gains for any two nontouching loops.

F_k: Gain of the k-th forward path

Δ_k: The resulting Δ after the k-th forward path is removed from the graph.

Example:

Catastrophic Generator Matrix

1. A catastrophic matrix maps information sequences with infinite Hamming weight to code sequences with finite Hamming weight.
2. For a catastrophic code, a finite number of transmission errors can cause an infinite number of errors in the decoded information sequence. Hence, theses codes should be avoided in practice.
3. Systematic generator matrices are never catastrophic.
4. It can be shown that a loop in the state diagram that produces a zero output for a non-zero input is a necessary and sufficient condition for a catastrophic matrix.

The (111) state has a loop associated with 1/00 in the following example: