CHAPTER 9

Trellises for Linear Block Codes

Constructing and representing codes with graphs have long been interesting problems to many coding theorists. The most commonly known graphical representation of a code is the trellis representation. A code *trellis diagram* is simply an edge-labeled directed graph in which every path represents a codeword (or a code sequence for a convolutional code). This representation makes it possible to implement maximum likelihood decoding (MLD) of a code with a significant reduction in decoding complexity. Trellis representation was first introduced by Forney [1] in 1973 as a means of explaining the decoding algorithm for convolutional codes devised by Viterbi [2]. This representation, together with the Viterbi decoding algorithm, has resulted in a wide range of applications of convolutional codes for error control in digital communications.

Trellis representation of linear block codes was first presented by Bahl, Cocke, Jelinek, and Raviv [3] in 1974. The first serious study of trellis structure and trellis construction of linear codes was due to Wolf. In his 1978 paper [4], Wolf presented the first method for constructing trellises for linear block codes and proved that an n-section trellis diagram of a q-ary (n, k) linear block code has at most $q^{\min\{k,n-k\}}$ states. Right after Wolf's work, Massey presented a simple but elegant paper [5] in which he gave a precise definition of a code trellis, derived some fundamental properties, and provided some implications of the trellis structure for encoding and decoding of codes; however, these early works in trellis representation of linear block codes did not arouse much enthusiasm, and for the next 10 years, there was basically no research and development in this area. It was Forney's paper in 1988 [6] that aroused enthusiasm for research in trellis structure of linear block codes. In this paper, Forney showed that some block codes, such as RM codes and some lattice codes, have relatively simple trellis structures. Motivated by Forney's work and the desire to achieve MLD for linear block codes to improve error performance over traditional hard-decision algebraic decoding, researchers made significant efforts to study the trellis structure and to devise trellis-based decoding algorithms for linear block codes. Developments have been dramatic and rapid, and new results are exciting.

This chapter gives an introductory coverage of trellises for linear block codes with a simple and unified approach. For further study, the readers are referred to the bibliography at the end of this chapter.

9.1 FINITE-STATE MACHINE MODEL AND TRELLIS REPRESENTATION OF A CODE

Any encoder for a code C, block or convolutional, has finite memory that stores information of the past. Suppose the information symbols are shifted into the encoder serially, and the coded symbols are shifted out of the encoder serially. As the information symbols are shifted into the encoder they are stored in the memory for a certain finite period of time (or interval). The stored information symbols in the memory affect the output code symbols. Let $\Gamma = \{0, 1, 2, \ldots\}$ denote the entire

encoding interval (or span) that consists of a sequence of encoding time instants. The interval between two consecutive time instants is defined as a *unit encoding interval*. During this unit encoding interval, code symbols are generated at the output of the encoder based on the current input information symbols and the past information symbols that are stored in the memory, according to a certain encoding rule. Therefore, the information symbols stored in the memory at any encoding time define a specific *state* of the encoder at that time instant. More precisely, the state of the encoder at time-i is defined by those information symbols, stored in the memory at time-i, that affect the current output code symbols during the interval from time-i to time-$(i+1)$ and future output code symbols. Because the memory has finite size, the compositions of the stored information symbols are finite; that is, at any time instant, the encoder has only a finite number of allowable states. As new information symbols are shifted into the memory some old information symbols may be shifted out of the encoder, and there is a transition from one state to another state—a *state transition*. With these definitions of a state and a state transition, the encoder can be modeled as a *finite-state machine* (*finite automata*), as shown in Figure 9.1. Then, the dynamic behavior of the encoder can be graphically represented by a state diagram in time, called a *trellis diagram* (or simply trellis), as shown in Figure 9.2, that consists of levels of nodes and edges connecting the nodes of one level to the nodes of the next level.

At time-0, the encoder starts from some specific initial state, denoted s_0. At time-i, the encoder resides in one and only one allowable state in a finite set. In the trellis diagram, the set of allowable states at time-i is represented by a set of nodes (or vertices) at the ith level, one for each allowable state. Hereafter, we use the terms *state, node*, and *vertex* interchangeably. The encoder moves from one allowable state at one time instant to another allowable state at the next time instant in one unit of time. This state transition, in the trellis diagram, is represented by a directed edge (commonly called a *branch*) connecting the starting state to the destination state. Each branch is labeled with the code symbols that are generated during the state transition. Therefore, each branch has a *label*. The set of allowable states at a given time instant i is called the *state space* of the encoder at time-i, denoted by $\Sigma_i(C)$. A state $s_i \in \Sigma_i(C)$ is said to be *reachable* if there exists an information sequence that takes (drives) the encoder from the initial state s_0 to state s_i at time-i. Every state of the encoder is reachable from the initial state s_0. In the trellis, every node at level-i for $i \in \Gamma$ is connected by a *path* (defined as a sequence of connected

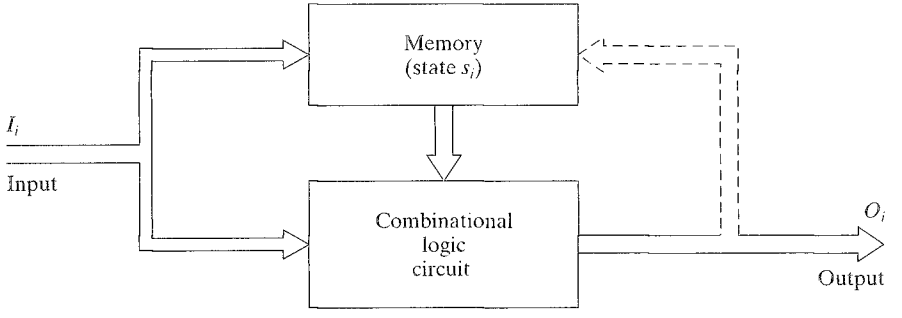

FIGURE 9.1: A finite-state machine model for an encoder with finite memory.

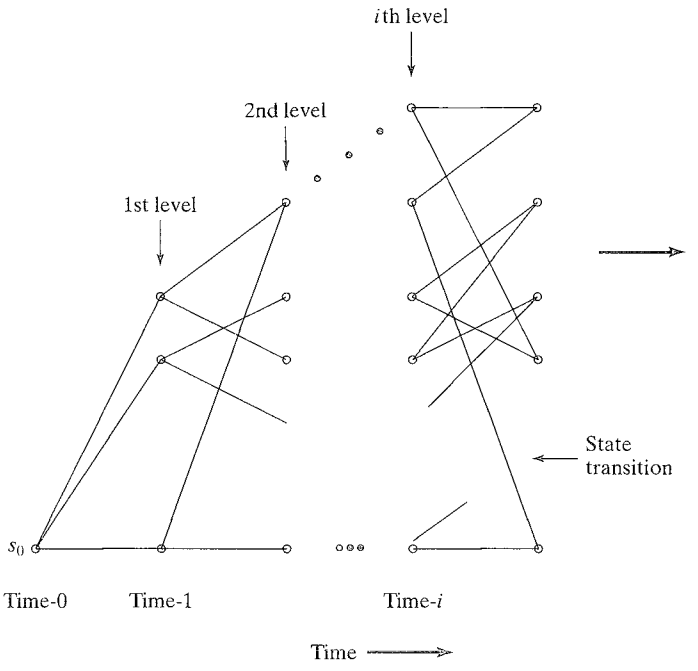

FIGURE 9.2: Trellis representation of a finite-state encoder.

branches) from the initial node. The label sequence of this path is a code sequence (or a prefix of a code sequence). Every node in the trellis has at least one incoming branch except for the initial node and at least one outgoing branch except for a node that represents the *final state* of the encoder at the end of the entire encoding interval. Encoding of an information sequence is equivalent to tracing a path in the trellis starting from the initial node s_0. If the encoding interval is semi-infinite, the trellis continues indefinitely; otherwise it terminates at a final node, denoted by s_f. Convolutional codes have semi-infinite trellises (to be discussed in Chapter 12), whereas the trellises for block codes terminate at the end of the encoding interval. In this graphical representation of a code, there is a one-to-one correspondence between a codeword (or code sequence) and a path in the code trellis; that is, every codeword is represented by a path in the code trellis, and conversely, every path in the code trellis represents a codeword. This trellis representation of a code makes it possible to implement MLD with a significant reduction in decoding complexity. This topic will be discussed in later chapters on trellis-based soft-decision decodings.

For $i \in \Gamma$, let I_i and O_i denote the input information block and its corresponding output code block, respectively, during the interval from time-i to time-$(i+1)$. Then, the dynamic behavior of the encoder for a linear code is governed by two functions:

1. **Output function:**

$$O_i = f_i(s_i, I_i),$$

where $f_i(s_i, I_i) \neq f_i(s_i, I_i')$ for $I_i \neq I_i'$.

2. **State transition function:**

$$s_{i+1} = g_i(s_i, I_i),$$

where $s_i \in \Sigma_i(C)$ and $s_{i+1} \in \Sigma_{i+1}(C)$ are called the current and next states, respectively.

In the trellis diagram for C, the current and next states are connected by an edge (s_i, s_{i+1}) labeled with O_i.

A code trellis is said to be *time-invariant* if there exists a finite period $\{0, 1, \ldots, \nu\} \subset \Gamma$ and a state space $\Sigma(C)$ such that

1. $\Sigma_i(C) \subset \Sigma(C)$ for $0 \leq i < \nu$, and $\Sigma_i(C) = \Sigma(C)$ for $i \geq \nu$, and
2. $f_i = f$ and $g_i = g$ for all $i \in \Gamma$.

A code trellis that is not time-invariant is said to be *time-varying*. A trellis diagram for a block code is, in general, time-varying; however, a trellis diagram for a convolutional code is usually time-invariant. Figures 9.3 and 9.4 depict a time-varying trellis diagram for a block code and a time-invariant trellis diagram for a convolutional code, respectively.

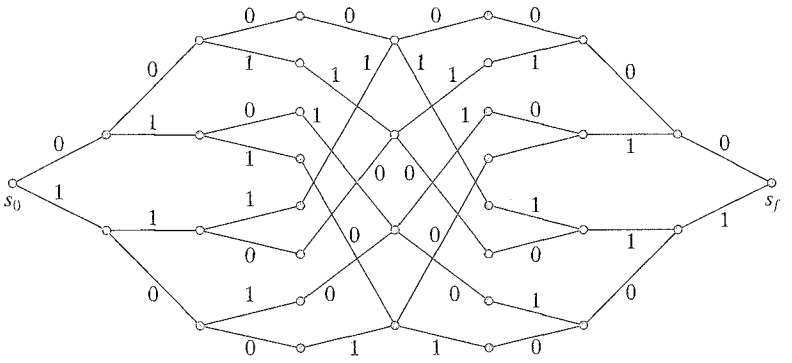

FIGURE 9.3: A time-varying trellis diagram for a block code.

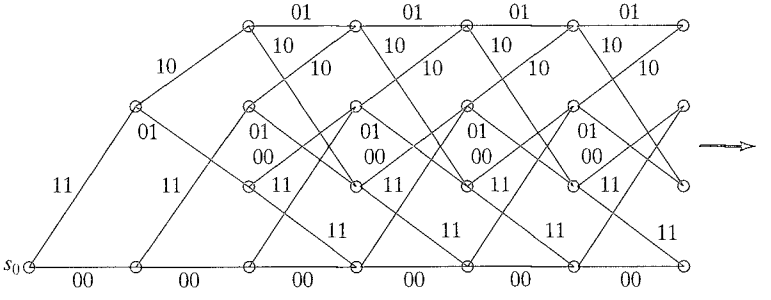

FIGURE 9.4: A time-invariant trellis diagram for a convolutional code.

In this chapter we are concerned only with trellises for binary linear block codes. Trellises for convolutional codes will be discussed in Chapter 12.

9.2 BIT-LEVEL TRELLISES FOR BINARY LINEAR BLOCK CODES

In this section we develop the basic concepts, structural properties, and construction of the primitive bit-level trellises for binary linear block codes.

Consider a binary (n, k) linear block code C with generator and parity-check matrices, $\mathbf{G}$ and $\mathbf{H}$, respectively. During each encoding interval Γ, a message of k information bits is shifted into the encoder memory one bit at a time and encoded into a codeword of n code bits. The n code bits are formed and shifted onto the channel in n bit times. Therefore, the encoding span Γ is finite and consists of $n + 1$ time instants,

$$\Gamma = \{0, 1, 2, \ldots, n\}.$$

Let $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ be the codeword generated at the output of the encoder. The code bit v_i is generated during the bit interval from time-i to time-$(i + 1)$. We can represent C with an n-section trellis diagram over the time span Γ, in which every branch is labeled with a single code bit. Let $E(C)$ denote the encoder for C.

Definition 9.1 An n-section bit-level trellis diagram for a binary linear block code C of length n, denoted by T, is a directed graph consisting of $n + 1$ levels of nodes (called states) and branches (also called edges) such that:

1. For $0 \leq i \leq n$, the nodes at the ith level represent the states in the state space $\Sigma_i(C)$ of the encoder $E(C)$ at time-i. At time-0 (or the zeroth level) there is only one node, denoted by s_0, called the *initial node* (or state). At time-n (or the nth level), there is only one node, denoted by s_f (or s_n), called the *final node* (or state).
2. For $0 \leq i < n$, a branch in the section of the trellis T between the ith level and the $(i + 1)$th level (or time-i and time-$(i + 1)$) connects a state $s_i \in \Sigma_i(C)$ to a state $s_{i+1} \in \Sigma_{i+1}(C)$ and is labeled with a code bit v_i that represents the encoder output in the bit interval from time-i to time-$(i + 1)$. A branch represents a state transition.
3. Except for the initial node, every node has at least one, but no more than two, incoming branches. Except for the final node, every node has at least one, but no more than two, outgoing branches. The initial node has no incoming branches. The final node has no outgoing branches. Two branches diverging from the same node have different labels; they represent two different transitions from the same starting state.
4. There is a directed path from the initial node s_0 to the final node s_f with a label sequence $(v_0, v_1, \ldots, v_{n-1})$ if and only if $(v_0, v_1, \ldots, v_{n-1})$ is a codeword in C.

Two states in the code trellis are said to be *adjacent* if they are connected by a branch. During one encoding interval Γ, the encoder starts from the initial state s_0, passes through a sequence of states

$$(s_0, s_1, \ldots, s_i, \ldots, s_f),$$

generates a code sequence

$$(v_0, v_1, \ldots, v_i, \ldots, v_{n-1}),$$

and then reaches the final state s_f. In the trellis, this sequence is represented by a path connecting the initial node s_0 to the final node s_f.

Figure 9.3 depicts the 8-section bit-level trellis diagram for the (8, 4) RM code of minimum distance 4.

For $0 \leq i \leq n$, let $|\Sigma_i(C)|$ denote the cardinality of the state space $\Sigma_i(C)$. Then, the sequence

$$(|\Sigma_0(C)|, |\Sigma_1(C)|, \ldots, |\Sigma_i(C)|, \ldots |\Sigma_n(C)|)$$

is called the *state space complexity profile*, which is a measure of the state complexity of the n-section bit-level code trellis T. We will prove later that for $0 \leq i \leq n$, $|\Sigma_i(C)|$ is a power of 2. We define

$$\rho_i(C) \triangleq \log_2 |\Sigma_i(C)|.$$

which is called the *state space dimension* at time-i. When there is no confusion, we simply use ρ_i for $\rho_i(C)$ for simplicity. The sequence

$$(\rho_0, \rho_1, \ldots, \rho_n)$$

is called the *state space dimension profile*. Because $|\Sigma_0(C)| = |\Sigma_n(C)| = 1$, $\rho_0 = \rho_n = 0$. From Figure 9.3 we find that the state space complexity profile and the state space dimension profile for the (8, 4) RM code are (1, 2, 4, 8, 4, 8, 4, 2, 1) and (0, 1, 2, 3, 2, 3, 2, 1, 0), respectively.

To facilitate the code trellis construction, we arrange the generator matrix $\mathbb{G}$ in a special form. Let $\mathbb{v} = (v_0, v_2, \ldots, v_{n-1})$ be a nonzero binary n-tuple. The first nonzero component of $\mathbb{v}$ is called the *leading* 1 of $\mathbb{v}$, and the last nonzero component of $\mathbb{v}$ is called the *trailing* 1 of $\mathbb{v}$. For example, the leading and trailing 1's of the 8-tuple (0, 1, 0, 1, 0, 1, 0, 1, 0) are v_1 and v_6. A generator matrix $\mathbb{G}$ for C is said to be in *trellis oriented form (TOF)* if the following two conditions hold:

1. The leading 1 of each row of $\mathbb{G}$ appears in a column before the leading 1 of any row below it.
2. No two rows have their trailing 1's in the same column.

Any generator matrix for C can be put in TOF by two steps of Gaussian elimination (or elementary row operations). Note that a generator matrix in TOF is not necessarily in systematic form.

EXAMPLE 9.1

Consider the first-order RM code of length 8, RM(1, 3). It is an (8, 4) code with the following generator matrix:

$$\mathbb{G} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

It is not in TOF. By interchanging the second and the fourth rows, we have

$$\mathbb{G}' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

We add the fourth row of the matrix to the first, second, and third rows. These additions result in the following matrix in TOF:

$$\mathbb{G}_{TOGM} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \mathbb{g}_2 \\ \mathbb{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix},$$

where *TOGM* stands for trellis oriented generator matrix.

In the following development of trellises for codes, we need to establish the correspondence between digit (or bit) positions and encoding time instants in $\Gamma = \{0, 1, 2, \ldots n\}$, as shown in Figure 9.5. For a codeword $\mathbb{v} = (v_0, v_1, \ldots, v_{n-1})$, the ith code digit v_i is generated in the interval between the time instant i and the time instant $i + 1$. Therefore, the ith bit position corresponds to time-i and time-$(i + 1)$.

Let $\mathbb{g} = (g_0, g_1, \ldots, g_{n-1})$ be a row in $\mathbb{G}_{TOGM}$ for code C. Let $\phi(\mathbb{g}) = \{i, i + 1, \ldots, j\}$ denote the smallest index interval that contains all the nonzero components of $\mathbb{g}$. This says that $g_i = 1$ and $g_j = 1$, and they are the leading and trailing 1's of $\mathbb{g}$, respectively. This interval $\phi(\mathbb{g}) = \{i, i + 1, \ldots, j\}$ is called the *digit* (or *bit*) *span* of $\mathbb{g}$. From Figure 9.5 we see that $\phi(\mathbb{g})$ occupies the time span from time-i to time-$(j + 1)$; that is, $\{i, i + 1, \ldots, j + 1\}$. In terms of time, we define the *time span* of $\mathbb{g}$, denoted by $\tau(\mathbb{g})$, as the following time interval:

$$\tau(\mathbb{g}) \triangleq \{i, i + 1, \ldots, j + 1\}. \tag{9.1}$$

For simplicity, we write $\phi(\mathbb{g}) = [i, j]$, and $\tau(\mathbb{g}) = [i, j + 1]$. Next, we define the *active time span* of $\mathbb{g}$, denoted by $\tau_a(\mathbb{g})$, as the time interval

$$\tau_a(\mathbb{g}) \triangleq \begin{cases} [i + 1, j], & \text{for } j > i \\ \phi(\text{empty set}), & \text{for } j = i. \end{cases} \tag{9.2}$$

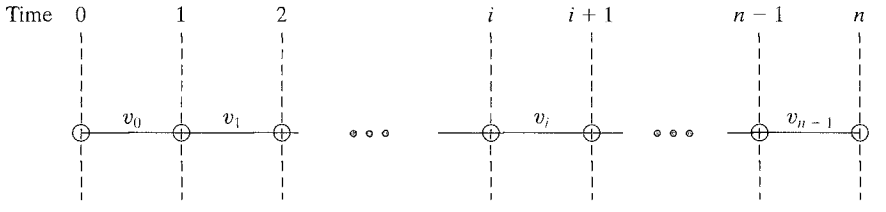

FIGURE 9.5: Time instants and code digit positions.

EXAMPLE 9.2

Consider the TOGM of the (8, 4) RM code given in Example 9.1.

We find that the bit spans of the rows are $\phi(\mathbb{g}_0) = [0, 3]$, $\phi(\mathbb{g}_1) = [1, 6]$, $\phi(\mathbb{g}_2) = [2, 5]$, and $\phi(\mathbb{g}_3) = [4, 7]$. The time spans of the rows are $\tau(\mathbb{g}_0) = [0, 4]$, $\tau(\mathbb{g}_1) = [1, 7]$, $\tau(\mathbb{g}_2) = [2, 6]$, and $\tau(\mathbb{g}_3) = [4, 8]$. The active time spans of the rows are $\tau_a(\mathbb{g}_0) = [1, 3]$, $\tau_a(\mathbb{g}_1) = [2, 6]$, $\tau_a(\mathbb{g}_2) = [3, 5]$, and $\tau_a(\mathbb{g}_3) = [5, 7]$.

$$
\begin{array}{l}
\text{Time} \longrightarrow \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \\
\mathbb{G}_{TOGM} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \mathbb{g}_2 \\ \mathbb{g}_3 \end{bmatrix} = \left[\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right] \\
\text{Bit position} \longrightarrow \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7
\end{array}
$$

Let $\mathbb{g}_0, \mathbb{g}_1, \ldots, \mathbb{g}_{k-1}$ be the k rows of a TOGM $\mathbb{G}_{TOGM}$ for an (n, k) linear code C with

$$\mathbb{g}_l = (g_{l0}, g_{l1}, \ldots, g_{l,n-1})$$

for $0 \leq l < k$. Then

$$\mathbb{G}_{TOGM} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \vdots \\ \mathbb{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{00} & g_{01} & g_{02} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & g_{12} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & g_{k-1,2} & \cdots & g_{k-1,n-1} \end{bmatrix}. \tag{9.3}$$

Let $(a_0, a_1, \ldots, a_{k-1})$ be the message of k information bits to be encoded. From (9.3) (or (3.3)), we find that the corresponding codeword is given by

$$\begin{aligned} \mathbb{v} &= (v_0, v_1, \ldots, v_{n-1}) \\ &= (a_0, a_1, \ldots, a_{k-1}) \cdot \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \vdots \\ \mathbb{g}_{k-1} \end{bmatrix} \\ &= a_0 \cdot \mathbb{g}_0 + a_1 \cdot \mathbb{g}_1 + \cdots + a_{k-1} \cdot \mathbb{g}_{k-1}. \end{aligned} \tag{9.4}$$

We see that the lth information bit a_l affects the output codeword $\mathbb{v}$ of the encoder $E(C)$ over the bit span $\phi(\mathbb{g}_l)$ of the lth row $\mathbb{g}_l$ of the TOGM $\mathbb{G}_{TOGM}$. Suppose $\phi(\mathbb{g}_l) = [i, j]$. Then, the information bit a_l affects the output code bits

$v_i, v_{i+1}, \cdots, v_j$ from time-i to time-$(j+1)$. At time-i, the input information bit a_l is regarded as the current input. At time-$(i+1)$, a_l is shifted into the encoder memory and remains in the memory for $j-i$ units of time. At time-$(j+1)$, it will have no effect on the future output code bits of the encoder, and it is shifted out of the encoder memory.

Now, we give a mathematical formulation of the state space of the n-section bit-level trellis for an (n, k) linear code C over $GF(2)$ with a TOGM $\mathbb{G}_{TOGM}$.

At time-i, $0 \leq i \leq n$, we divide the rows of $\mathbb{G}_{TOGM}$ into three disjoint subsets:

1. $\mathbb{G}_i^p$ consists of those rows of $\mathbb{G}_{TOGM}$ whose bit spans are contained in the interval $[0, i-1]$.
2. $\mathbb{G}_i^f$ consists of those rows of $\mathbb{G}_{TOGM}$ whose bit spans are contained in the interval $[i, n-1]$.
3. $\mathbb{G}_i^s$ consists of those rows of $\mathbb{G}_{TOGM}$ whose active time spans contain time-i.

EXAMPLE 9.3

Consider the TOGM $\mathbb{G}_{TOGM}$ of the (8, 4) RM code given in Example 9.2. At time-2, we find that

$$\mathbb{G}_2^p = \phi, \quad \mathbb{G}_2^f = \{\mathbb{g}_2, \mathbb{g}_3\}, \quad \mathbb{G}_2^s = \{\mathbb{g}_0, \mathbb{g}_1\},$$

where ϕ denotes the empty set. At time-4, we find that

$$\mathbb{G}_4^p = \{\mathbb{g}_0\}, \quad \mathbb{G}_4^f = \{\mathbb{g}_3\}, \quad \mathbb{G}_4^s = \{\mathbb{g}_1, \mathbb{g}_2\}.$$

Let A_i^p, A_i^f, and A_i^s denote the subsets of information bits, $a_0, a_1, \cdots, a_{k-1}$, that correspond to the rows of $\mathbb{G}_i^p$, $\mathbb{G}_i^f$, and $\mathbb{G}_i^s$, respectively (see (9.4) for the correspondence). The information bits in A_i^p do not affect the encoder outputs after time-i, and hence they become the *past* with respect to time-i. The information bits in A_i^f affect the encoder outputs only after time-i. Because the active time spans of the rows in $\mathbb{G}_i^s$ contain the time instant i, the information bits in A_i^s affect not only the past encoder outputs up to time-i but also the future encoder outputs beyond time-i. Therefore, the bits in A_i^s are the information bits stored in the encoder memory that affect the current output code bit v_i and the future output code bits beyond time-i. These information bits in A_i^s hence define a state of the encoder $E(C)$ for the code C at time-i. Let

$$\rho_i \triangleq |A_i^s| = |\mathbb{G}_i^s|.$$

Then, there are 2^{ρ_i} distinct states in which the encoder $E(C)$ can reside at time-i; each state is defined by a specific combination of the ρ_i information bits in A_i^s. These states form the state space $\Sigma_i(C)$ of the encoder $E(C)$ (or simply of the code C). The parameter ρ_i is hence the dimension of the state space $\Sigma_i(C)$. In the trellis representation of C, the states in $\Sigma_i(C)$ are represented by 2^{ρ_i} nodes at the ith level of the trellis. The set A_i^s is called the *state defining information set* at time-i.

From the preceding analysis we see that at time-i for $0 \leq i \leq n$, the dimension ρ_i of the state space $\Sigma_i(C)$ is simply equal to the number of rows in the TOGM

TABLE 9.1: Partition of the TOGM of the (8, 4) RM code.

Time i	$\mathbb{G}_i^p$	$\mathbb{G}_i^f$	$\mathbb{G}_i^s$	ρ_i
0	ϕ	$\{\mathbb{g}_0, \mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$	ϕ	0
1	ϕ	$\{\mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$	$\{\mathbb{g}_0\}$	1
2	ϕ	$\{\mathbb{g}_2, \mathbb{g}_3\}$	$\{\mathbb{g}_0, \mathbb{g}_1\}$	2
3	ϕ	$\{\mathbb{g}_3\}$	$\{\mathbb{g}_0, \mathbb{g}_1, \mathbb{g}_2\}$	3
4	$\{\mathbb{g}_0\}$	$\{\mathbb{g}_3\}$	$\{\mathbb{g}_1, \mathbb{g}_2\}$	2
5	$\{\mathbb{g}_0\}$	ϕ	$\{\mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$	3
6	$\{\mathbb{g}_0, \mathbb{g}_2\}$	ϕ	$\{\mathbb{g}_1, \mathbb{g}_3\}$	2
7	$\{\mathbb{g}_0, \mathbb{g}_1, \mathbb{g}_2\}$	ϕ	$\{\mathbb{g}_3\}$	1
8	$\{\mathbb{g}_0, \mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$	ϕ	ϕ	0

$\mathbb{G}_{TOGM}$ of C whose active time spans contain time-i. We say that these rows are *active* at time-i. Therefore, from the TOGM $\mathbb{G}_{TOGM}$ we can easily determine the state space dimension profile $(\rho_0, \rho_1, \cdots, \rho_n)$ simply by counting the number of rows in $\mathbb{G}_{TOGM}$ that are active at time-i for $0 \leq i \leq n$.

EXAMPLE 9.4

Again consider the TOGM $\mathbb{G}_{TOGM}$ of the (8, 4) RM code given in Example 9.2. The partition of the TOGM at each time instant in $\Gamma = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ is given in Table 9.1. Counting the number of rows of $\mathbb{G}_{TOGM}$ that are active at time-i for $0 \leq i \leq 8$, we obtain the state space dimension profile (0, 1, 2, 3, 2, 3, 2, 1, 0) for the 8-section trellis of the (8, 4) RM code.

For $0 \leq i < n$, suppose the encoder $E(C)$ is in state $s_i \in \Sigma_i(C)$. From time-i to time-$(i + 1)$, $E(C)$ generates a code bit v_i and moves from state s_i to a state $s_{i+1} \in \Sigma_{i+1}(C)$. Let

$$\mathbb{G}_i^s = \{\mathbb{g}_1^{(i)}, \mathbb{g}_2^{(i)}, \ldots, \mathbb{g}_{\rho_i}^{(i)}\} \tag{9.5}$$

and

$$A_i^s = \{a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}\}, \tag{9.6}$$

where $\rho_i = |\mathbb{G}_i^s| = |A_i^s|$. The current state s_i of the encoder is defined by a specific combination of the information bits in A_i^s.

Let $\mathbb{g}^*$ denote the row in $\mathbb{G}_i^f$ whose leading 1 is at bit position i. The uniqueness of this row $\mathbb{g}^*$ (if it exists) is guaranteed by the first condition in the definition of a generator matrix in TOF. Let g_i^* denote the ith component of $\mathbb{g}^*$. Then, $g_i^* = 1$. Let a^* denote the information bit that corresponds to row $\mathbb{g}^*$. It follows from (9.3) and (9.4) and the structure of the TOGM $\mathbb{G}_{TOGM}$ that the output code bit v_i generated during the bit interval between time-i and time-$(i + 1)$ is given by

$$v_i = a^* + \sum_{l=1}^{\rho_i} a_l^{(i)} \cdot g_{l,i}^{(i)}, \tag{9.7}$$

where $g_{l,i}^{(i)}$ is the ith component of $\mathbf{g}_l^{(i)}$ in $\mathbb{G}_i^s$. Note that the information bit a^* begins to affect the output of the encoder $E(C)$ at time-i. For this reason, bit a^* is regarded as the *current input information bit*. The second term in (9.7) is the contribution from the state s_i defined by the information bits in $A_i^s = \{a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}\}$ that are stored in memory. From (9.7) we see that the current output v_i is uniquely determined by the current state s_i of the encoder $E(C)$ and the current input information bit a^*. The output code bit v_i can have two possible values depending on the current input information bit a^*; each value takes the encoder $E(C)$ to a different state at time-$(i+1)$; that is, there are two possible transitions from the current state s_i to two states in $\Sigma_{i+1}(C)$ at time-$(i+1)$. In the code trellis T for C, there are two branches diverging from the node s_i, labeled with 0 and 1, respectively.

Suppose there is no such row $\mathbf{g}^*$ in $\mathbb{G}_i^f$. Then, the output code bit v_i at time-i is given by

$$v_i = \sum_{l=1}^{\rho_i} a_l^{(i)} \cdot g_{l,i}^{(i)}. \tag{9.8}$$

In this case we may regard that the current input information bit a^* is being set to 0; that is, $a^* = 0$ (this is called a *dummy* information bit). The output code bit v_i can take only one value given by (9.8), and there is only one possible transition from the current state s_i to a state in $\Sigma_{i+1}(C)$. In the trellis T, there is only one branch diverging from the node s_i.

From the preceding analysis we see that the output function of the encoder $E(C)$ is given by either (9.7) or (9.8). This result shows the time-varying nature of the output function of a linear block code encoder and hence the time-varying nature of the code trellis.

EXAMPLE 9.5

Consider the (8, 4) RM code with its TOGM $\mathbb{G}_{TOGM}$ given in Example 9.2. From Table 9.1 we find that at time-2, $\mathbb{G}_2^p = \phi$, $\mathbb{G}_2^f = \{\mathbf{g}_2, \mathbf{g}_3\}$, and $\mathbb{G}_2^s = \{\mathbf{g}_0, \mathbf{g}_1\}$. Therefore, $A_2^s = \{a_0, a_1\}$, the information bits a_0 and a_1 define the state of the encoder at time-2, and there are four distinct states defined by four combinations of values of a_0 and a_1, $\{00, 01, 10, 11\}$. We also find that $\mathbf{g}^* = \mathbf{g}_2$. Hence, the current input information bit at time-2 is $a^* = a_2$. The current output code bit v_2 is given by

$$\begin{aligned} v_2 &= \boxed{a_2} + a_0 \cdot g_{02} + a_1 \cdot g_{12} \\ &= \boxed{a_2} + a_0 \cdot 1 + a_1 \cdot 0 \\ &= \boxed{a_2} + a_0, \end{aligned}$$

where $\boxed{a_2}$ denotes the current input. For every state defined by a_0 and a_1, v_2 has two possible values depending on a_2. In the trellis there are two branches diverging from each state at time-2, as shown in Figure 9.3. Now, consider time-3. For $i = 3$, we find that $\mathbb{G}_3^p = \phi$, $\mathbb{G}_3^f = \{\mathbf{g}_3\}$, and $\mathbb{G}_3^s = \{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2\}$. Therefore, $A_3^s = \{a_0, a_1, a_2\}$, and the information bits in A_3^s define eight states at time-3, as shown in Figure 9.3. There is no row $\mathbf{g}^*$ in $\mathbb{G}_3^f$ with leading 1 at bit position $i = 3$. Hence, we set the

current input information bit $a^* = 0$. The output code bit v_3 is given by (9.8),

$$\begin{aligned} v_3 &= a_0 \cdot g_{03} + a_1 \cdot g_{13} + a_2 \cdot g_{23} \\ v_3 &= a_0 \cdot 1 + a_1 \cdot 1 + a_2 \cdot 1 \\ &= a_0 + a_1 + a_2. \end{aligned}$$

In the trellis, there is only one branch diverging from each of the eight states, as shown in Figure 9.3.

So far, we have formulated the state space and determined the output function of a linear block code encoder $E(C)$. Next we need to determine the state transition of the encoder from one time instant to another.

Let $\mathbb{g}^0$ denote the row in $\mathbb{G}_i^s$ whose trailing 1 is at bit position-i, that is, the ith component $\mathbb{g}_i^0$ of $\mathbb{g}^0$ is the last nonzero component of $\mathbb{g}^0$. (Note that this row $\mathbb{g}^0$ may not exist.) The uniqueness of the row $\mathbb{g}^0$ (if it exists) is guaranteed by the second condition of a generator matrix in TOF. Let a^0 be the information bit in A_i^s that corresponds to row $\mathbb{g}^0$. Then, at time-$(i+1)$,

$$\mathbb{G}_{i+1}^s = (\mathbb{G}_i^s \backslash \{\mathbb{g}^0\}) \cup \{\mathbb{g}^*\} \tag{9.9}$$

and

$$A_{i+1}^s = (A_i^s \backslash \{a^0\}) \cup \{a^*\}. \tag{9.10}$$

The information bits in A_{i+1}^s define the state space $\Sigma_{i+1}(C)$ at time-$(i+1)$. The change from A_i^s to A_{i+1}^s defines the state transition of the encoder $E(C)$ from the current state s_i defined by A_i^s to the next state s_{i+1} defined by A_{i+1}^s. We may regard the information bit a^0 as the *oldest* information bit stored in the encoder memory at time-i. As the encoder $E(C)$ moves from time-i to time-$(i+1)$, a^0 is shifted out of the memory and a^* (if it exists) is shifted into the memory.

The state defining sets A_i^s and A_{i+1}^s, and the output functions given by (9.7) and (9.8), provide all the information needed to construct the n-section bit-level trellis diagram T for the (n, k) code C.

The construction of the n-section trellis T is carried out serially, section by section. Suppose the trellis has been constructed up to section-i (i.e., from time-0 to time-i). Now, we want to construct the $(i+1)$th section from time-i to time-$(i+1)$. The state space $\Sigma_i(C)$ is known. The $(i+1)$th section is constructed as follows:

1. Determine $\mathbb{G}_{i+1}^s$ and A_{i+1}^s from (9.9) and (9.10). Form the state space $\Sigma_{i+1}(C)$ at time-$(i+1)$.
2. For each state $s_i \in \Sigma_i(C)$, determine its state transition(s) based on the change from A_i^s to A_{i+1}^s. Connect s_i to its adjacent state(s) in $\Sigma_{i+1}(C)$ by branches.
3. For each state transition, determine the output code bit v_i from the output function of (9.7) or (9.8), and label the corresponding branch in the trellis with v_i.

EXAMPLE 9.6

Consider the (8, 4) RM code with its TOGM $\mathbb{G}_{TOGM}$ given in Example 9.2. Suppose the trellis for this code has been constructed up to time-4. At this point, we find from Table 9.1 that $\mathbb{G}_4^s = \{\mathbb{g}_1, \mathbb{g}_2\}$. The state space $\Sigma_4(C)$ is defined by $A_4^s = \{a_1, a_2\}$. There are four states at time-4, which are determined by the four combinations of a_1 and a_2: $\{(0, 0), (0, 1), (1, 0), (1, 1)\}$. To construct the trellis section from time-4 to time-5, we find that there is no row $\mathbb{g}^0$ in $\mathbb{G}_4^s = \{\mathbb{g}_1, \mathbb{g}_2\}$, but there is a row $\mathbb{g}^*$ in $\mathbb{G}_4^f = \{\mathbb{g}_3\}$, which is $\mathbb{g}_3$. Therefore, at time-5, we have

$$\mathbb{G}_5^s = \{\mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$$

and

$$A_5^s = \{a_1, a_2, a_3\}.$$

The state space $\Sigma_5(C)$ is then defined by the three bits in A_5^s. The eight states in $\Sigma_5(C)$ are defined by the eight combinations of a_1, a_2, and a_3: $\{(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)\}$. Suppose the current state s_4 at time-4 is defined by (a_1, a_2). Then, the next state at time-5 is either the state s_5 defined by $(a_1, a_2, a_3 = 0)$ or the state s_5' defined by $(a_1, a_2, a_3 = 1)$. The output code bit v_4 is given by

$$v_4 = \boxed{a_3} + a_1 \cdot 1 + a_2 \cdot 1,$$

which has two values depending on whether the current input bit a_3 is 0 or 1. Connecting each state at time-4 to its two adjacent states at time-5 by branches and labeling each branch with the corresponding code bit v_4 for either $a_3 = 0$ or $a_3 = 1$, we complete the trellis section from time-4 to time-5. To construct the next trellis section from time-5 to time-6, we first find that there is a row $\mathbb{g}^0$ in $\mathbb{G}_5^s = \{\mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$, which is $\mathbb{g}_2$, and there is no row $\mathbb{g}^*$ in $\mathbb{G}_5^f = \phi$. Therefore,

$$\mathbb{G}_6^s = \mathbb{G}_5^s \backslash \{\mathbb{g}_2\} = \{\mathbb{g}_1, \mathbb{g}_3\},$$

and

$$A_6^s = \{a_1, a_3\}.$$

From the change from A_5^s to A_6^s, we find that two states defined by $(a_1, a_2 = 0, a_3)$ and $(a_1, a_2 = 1, a_3)$ at time-5 move into the same state defined by (a_1, a_3) at time-6. The two connecting branches are labeled with

$$v_5 = \boxed{0} + a_1 \cdot 0 + (a_2 = 0) \cdot 1 + a_3 \cdot 1,$$

and

$$v_5 = \boxed{0} + a_1 \cdot 0 + (a_2 = 1) \cdot 1 + a_3 \cdot 1,$$

respectively, where $\boxed{0}$ denotes the dummy input. This complete the construction of the trellis section from time-5 to time-6. Continue with this construction process until the trellis terminates at time-8.

During the encoding interval $\Gamma = \{0, 1, \ldots, n\}$, the output function of the encoder $E(C)$ changes between (9.7) and (9.8). Also, the set $\{\mathbb{g}_{1,i}^{(i)}, \mathbb{g}_{2,i}^{(i)}, \ldots, \mathbb{g}_{\rho_i,i}^{(i)}\}$ in the summations of (9.7) and (9.8) may change from one time instant to another. This is because each column in the TOGM is, in general, not a downward shift of the column before it. Therefore, the output function of $E(C)$ is time-varying. Furthermore, from (9.10), we see that the size and the composition of the state-defining information set may change from one time instant to another depending on whether there is an oldest information bit to be shifted out of the memory and a current information bit to be shifted into the memory. As a result, the state space of the encoder $E(C)$ varies from time to time. Therefore, the trellis T is time-varying.

9.3 STATE LABELING

The construction of a code trellis can be facilitated by labeling the states at each level of the trellis. State labeling is also necessary in the implementation of a trellis-based decoding algorithm. Next, we present a labeling method based on the information set that defines the state space at a particular encoding time instant.

In a code trellis, each state is labeled by a fixed sequence (or given a name). This labeling can be accomplished by using a k-tuple $l(s)$ with components corresponding to the k information bits, $a_0, a_1, \ldots, a_{k-1}$, in a message. At time-$i$, all the components of $l(s)$ are set to zero except for the components at the positions corresponding to the information bits in $A_i^s = \{a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}\}$. Every combination of the ρ_i bits at the positions corresponding to the bits in A_i^s gives the label $l(s_i)$ for the state s_i defined by the information bits, $a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}$.

EXAMPLE 9.7

Consider the (8, 4) code given in Example 9.2. At time-4, the state-defining information set is $A_4^s = \{a_1, a_2\}$. There are four states corresponding to four combinations of a_1 and a_2. Therefore, the label for each of these four states is given by $(0, a_1, a_2, 0)$. At time-5, $A_5^s = \{a_1, a_2, a_3\}$, and there are eight states. The label for each of these eight states is given by $(0, a_1, a_2, a_3)$.

With the described state labeling, we can restate the trellis construction procedure. Suppose the trellis T has been constructed up to section-i. At this point, $\mathbb{G}_i^s$, A_i^s, and $\Sigma_i(C)$ are known. Each state $s_i \in \Sigma_i(C)$ is labeled by a k-tuple $l(s_i)$. The $(i+1)$th section is constructed as follows:

1. Determine $\mathbb{G}_{i+1}^s$ and A_{i+1}^s from (9.9) and (9.10).
2. Form the state space $\Sigma_{i+1}(C)$ at time-$(i+1)$ and label each state in $\Sigma_{i+1}(C)$ based on A_{i+1}^s. The states in $\Sigma_{i+1}(C)$ form the nodes of the code trellis T at the $(i+1)$th level.
3. For each state $s_i \in \Sigma_i(C)$ at time-i, determine its transition(s) to the state(s) in $\Sigma_{i+1}(C)$ based on the information bits a^* and a^0. For each transition from a state $s_i \in \Sigma_i(C)$ to a state $s_{i+1} \in \Sigma_{i+1}(C)$, connect the state s_i to the state s_{i+1} by a branch (s_i, s_{i+1}).
4. For each state transition (s_i, s_{i+1}), determine the output code bit v_i and label the branch (s_i, s_{i+1}) with v_i.

Recall that at time-i there are two branches diverging from a state in $\Sigma_i(C)$ if there exists a current information bit a^*. One branch corresponds to $a^* = 0$, and the other corresponds to $a^* = 1$. For the convenience of graphical representation in the code trellis T, we use the upper branch to represent $a^* = 0$ and the lower branch to represent $a^* = 1$. If a^* is a dummy information bit, then there is only one branch diverging from each state in $\Sigma_i(C)$. This single branch represents a dummy information bit. Using the described representation, we can easily extract the information bits from each path in the trellis (the dummy information bits are deleted).

EXAMPLE 9.8

Consider the state labeling and trellis construction for the (8, 4) RM code given in Example 9.2 whose TOGM $\mathbb{G}_{TOGM}$ is repeated here:

$$\mathbb{G}_{TOGM} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

For $0 \le i \le 8$, we determine the submatrix $\mathbb{G}_i^s$ and the state-defining information set A_i^s as listed in Table 9.2. From A_i^s we form the label for each state in $\Sigma_i(C)$ as shown in Table 9.2. The state transitions from time-i to time-$(i+1)$ are determined by the change from A_i^s to A_{i+1}^s. Following the trellis construction procedure given described, we obtain the 8-section trellis diagram for the (8, 4) RM code as shown in Figure 9.6. Each state in the trellis is labeled by a 4-tuple.

In many cases, we do not need k bits for labeling the states of the n-section bit-level trellis for a binary (n, k) linear block code C. Let $(\rho_0, \rho_1, \ldots, \rho_n)$ be the state space dimension profile of the trellis. We define

$$\rho_{\max}(C) \triangleq \max_{0 \le i \le n} \rho_i, \tag{9.11}$$

TABLE 9.2: State-defining sets and labels for the 8-section trellis for the (8, 4) RM code.

i	$\mathbb{G}_i^s$	a^*	a^0	A_i^s	State Label
0	ϕ	a_0	—	ϕ	(0000)
1	$\{\mathbf{g}_0\}$	a_1	—	$\{a_0\}$	(a_0000)
2	$\{\mathbf{g}_0, \mathbf{g}_1\}$	a_2	—	$\{a_0, a_1\}$	(a_0a_100)
3	$\{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2\}$	—	a_0	$\{a_0, a_1, a_2\}$	$(a_0a_1a_20)$
4	$\{\mathbf{g}_1, \mathbf{g}_2\}$	a_3	—	$\{a_1, a_2\}$	$(0a_1a_20)$
5	$\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3\}$	—	a_2	$\{a_1, a_2, a_3\}$	$(0a_1a_2a_3)$
6	$\{\mathbf{g}_1, \mathbf{g}_3\}$	—	a_1	$\{a_1, a_3\}$	$(0a_10a_3)$
7	$\{\mathbf{g}_3\}$	—	a_3	$\{a_3\}$	$(000a_3)$
8	ϕ	—	—	ϕ	(0000)

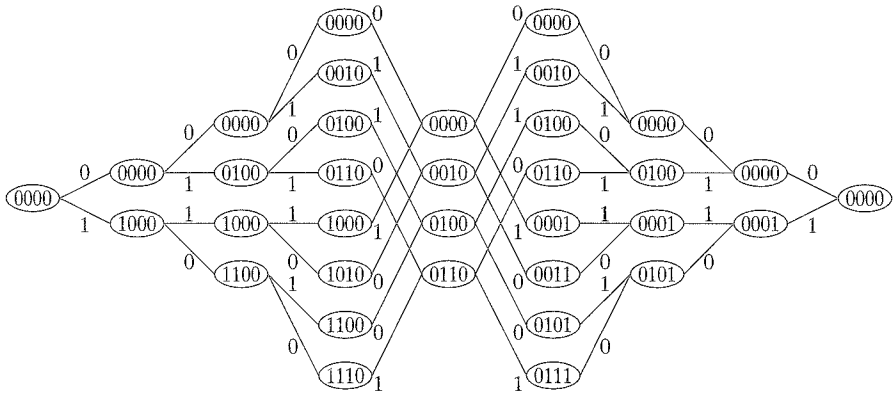

FIGURE 9.6: The 8-section trellis diagram for the (8, 4) RM code with state labeling by the state-defining information set.

which is simply the maximum state space dimension of the trellis. Because $\rho_i = |\mathbb{G}_i^s| = |A_i^s|$ for $0 \le i \le n$, and $\mathbb{G}_i^s$ is a submatrix of the generator matrix of C, we have

$$\rho_{\max}(C) \le k. \tag{9.12}$$

In general, $\rho_{\max}$ is smaller than k. Because the number of states at any level of the trellis is less than or at most equal to $2^{\rho_{\max}(C)}$, $\rho_{\max}(C)$ bits are sufficient for labeling the states in the trellis. Consider the state space $\Sigma_i(C)$ at time-i with $0 \le i \le n$ that is defined by the set $\{a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}\}$ of ρ_i information bits. For each state $s_i \in \Sigma_i(C)$, we form a $\rho_{\max}(C)$-tuple, denoted by $l(s_i)$, in which the first ρ_i components are simply $a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}$, and the remaining $\rho_{\max}(C) - \rho_i$ components are set to 0; that is,

$$l(s_i) \triangleq (a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}, 0, 0, \ldots, 0).$$

Then, $l(s_i)$ is the label for the state s_i.

EXAMPLE 9.9

For the (8, 4) RM code, the state space dimension profile of its 8-section trellis is (0, 1, 2, 3, 2, 3, 2, 1, 0) (see Figure 9.6). Hence, $\rho_{\max}(C) = 3$. Using 3 bits for labeling the states as described previously, we give the state labels in Table 9.3. Compared with the state labeling given in Example 9.8, 1 bit is saved.

States also can be labeled based on the parity-check matrix $\mathbb{H}$ of a code. In this case, $n-k$ bits are needed to label each state. Therefore, for high-rate codes with small $n - k$, labeling states based on the parity-check matrix is more efficient than labeling based on the TOGM $\mathbb{G}$ of the code. This topic will be discussed in a later section.

EXAMPLE 9.10

Consider the second-order RM code of length 16. It is a (16, 11) code with a minimum distance of 4. Its conventional generator matrix is given in Example 4.2.

TABLE 9.3: State labeling for the (8, 4) RM code using $\rho_{\max}(C) = 3$ bits.

i	A_i^s	State Label
0	ϕ	(000)
1	$\{a_0\}$	$(a_0 00)$
2	$\{a_0, a_1\}$	$(a_0 a_1 0)$
3	$\{a_0, a_1, a_2\}$	$(a_0 a_1 a_2)$
4	$\{a_1, a_2\}$	$(a_1 a_2 0)$
5	$\{a_1, a_2, a_3\}$	$(a_1 a_2 a_3)$
6	$\{a_1, a_3\}$	$(a_1 a_3 0)$
7	$\{a_3\}$	$(a_3 00)$
8	ϕ	(000)

Performing elementary row operations, we obtain the following TOGM:

$$\mathbb{G}_{TOGM} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \mathbb{g}_2 \\ \mathbb{g}_3 \\ \mathbb{g}_4 \\ \mathbb{g}_5 \\ \mathbb{g}_6 \\ \mathbb{g}_7 \\ \mathbb{g}_8 \\ \mathbb{g}_9 \\ \mathbb{g}_{10} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

From this TOGM, we easily find the state space dimension profile, (0, 1, 2, 3, 3, 4, 4, 4, 3, 4, 4, 4, 3, 3, 2, 1, 0). The maximum state space dimension is $\rho_{\max}(C) = 4$. Therefore, we can use 4 bits to label the trellis states. The state-defining information sets and state labels at each time instant are given in Table 9.4. With the information given in this table, we can readily construct the 16-section bit-level trellis for the (16, 11) RM code, as shown in Figure 9.7.

9.4 STRUCTURAL PROPERTIES OF BIT-LEVEL TRELLISES

In this section we develop some fundamental structural properties of an n-section bit-level trellis for an (n, k) binary linear block code C.

For $0 \leq i < j < n$, let $C_{i,j}$ denote the subcode of C consisting of those codewords in C whose nonzero components are confined to the span of $j - i$ consecutive positions in the set $\{i, i + 1, \ldots, j - 1\}$. Clearly, every codeword in $C_{i,j}$ is of the form

$$(\underbrace{0, 0, \ldots, 0}_{i}, v_i, v_{i+1}, \ldots, v_{j-1}, \underbrace{0, 0, \ldots, 0}_{n-j}),$$

and $C_{i,j}$ is a subcode of C. It follows from the definition of $C_{i,j}$ and the structure of the TOGM $\mathbb{G}_{TOGM}$ for C that $C_{i,j}$ is spanned by those rows in $\mathbb{G}_{TOGM}$ whose

TABLE 9.4: State-defining sets and labels for the 16-section trellis for the (16, 11) RM code.

i	$\mathbb{G}_i^s$	a^*	a^0	A_i^s	State Label
0	ϕ	a_0	—	ϕ	(0000)
1	$\{\mathbb{g}_0\}$	a_1	—	$\{a_0\}$	(a_0000)
2	$\{\mathbb{g}_0, \mathbb{g}_1\}$	a_2	—	$\{a_0, a_1\}$	(a_0a_100)
3	$\{\mathbb{g}_0, \mathbb{g}_1, \mathbb{g}_2\}$	a_3	a_0	$\{a_0, a_1, a_2\}$	$(a_0a_1a_20)$
4	$\{\mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$	a_4	—	$\{a_1, a_2, a_3\}$	$(a_1a_2a_30)$
5	$\{\mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3, \mathbb{g}_4\}$	a_5	a_2	$\{a_1, a_2, a_3, a_4\}$	$(a_1a_2a_3a_4)$
6	$\{\mathbb{g}_1, \mathbb{g}_3, \mathbb{g}_4, \mathbb{g}_5\}$	a_6	a_1	$\{a_1, a_3, a_4, a_5\}$	$(a_1a_3a_4a_5)$
7	$\{\mathbb{g}_3, \mathbb{g}_4, \mathbb{g}_5, \mathbb{g}_6\}$	—	a_4	$\{a_3, a_4, a_5, a_6\}$	$(a_3a_4a_5a_6)$
8	$\{\mathbb{g}_3, \mathbb{g}_5, \mathbb{g}_6\}$	a_7	—	$\{a_3, a_5, a_6\}$	$(a_3a_5a_60)$
9	$\{\mathbb{g}_3, \mathbb{g}_5, \mathbb{g}_6, \mathbb{g}_7\}$	a_8	a_6	$\{a_3, a_5, a_6, a_7\}$	$(a_3a_5a_6a_7)$
10	$\{\mathbb{g}_3, \mathbb{g}_5, \mathbb{g}_7, \mathbb{g}_8\}$	a_9	a_5	$\{a_3, a_5, a_7, a_8\}$	$(a_3a_5a_7a_8)$
11	$\{\mathbb{g}_3, \mathbb{g}_7, \mathbb{g}_8, \mathbb{g}_9\}$	—	a_7	$\{a_3, a_7, a_8, a_9\}$	$(a_3a_7a_8a_9)$
12	$\{\mathbb{g}_3, \mathbb{g}_8, \mathbb{g}_9\}$	a_{10}	a_3	$\{a_3, a_8, a_9\}$	$(a_3a_8a_90)$
13	$\{\mathbb{g}_8, \mathbb{g}_9, \mathbb{g}_{10}\}$	—	a_9	$\{a_8, a_9, a_{10}\}$	$(a_8a_9a_{10}0)$
14	$\{\mathbb{g}_8, \mathbb{g}_{10}\}$	—	a_8	$\{a_8, a_{10}\}$	$(a_8a_{10}00)$
15	$\{\mathbb{g}_{10}\}$	—	a_{10}	$\{a_{10}\}$	$(a_{10}000)$
16	ϕ	—	—	ϕ	(0000)

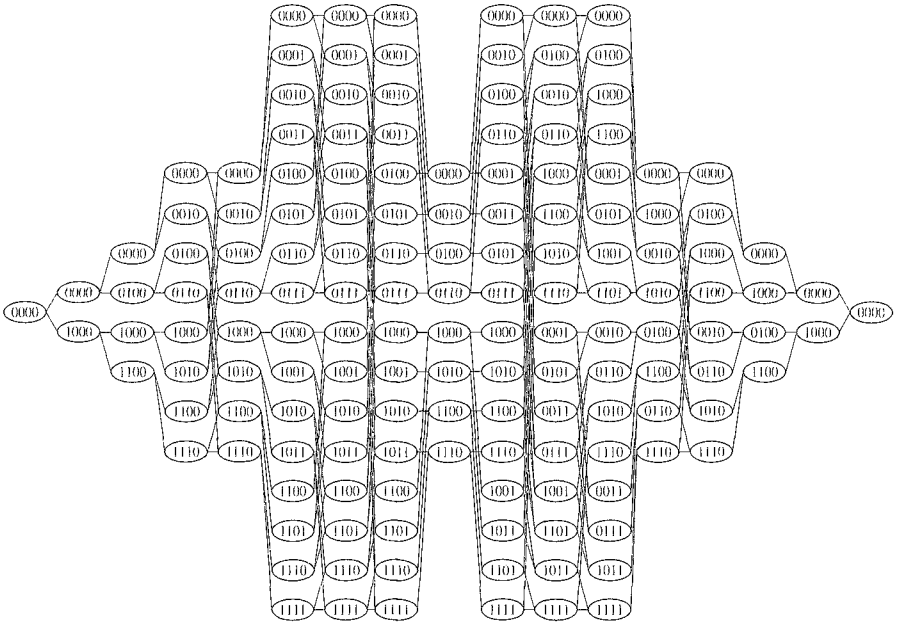

FIGURE 9.7: 16-section trellis for the (16, 11) RM code with state labeling by the state-defining information sets.

bit spans are contained in the interval $[i, j-1]$. The two subcodes $C_{0,i}$ and $C_{i,n}$ are spanned by the rows in $\mathbb{G}_i^p$ and $\mathbb{G}_i^f$, respectively, and they are called the *past* and *future* subcodes of C with respect to time-i.

For a linear code B, let $k(B)$ denote its dimension. Then, $k(C_{0,i}) = |\mathbb{G}_i^p|$, and $k(C_{i,n}) = |\mathbb{G}_i^f|$. Recall that the dimension of the state space $\Sigma_i(C)$ at time-i is $\rho_i(C) = |\mathbb{G}_i^s|$. Then, it follows from the definitions of $\mathbb{G}_i^s$, $\mathbb{G}_i^p$, and $\mathbb{G}_i^f$ that

$$\begin{aligned}\rho_i(C) &= k - |\mathbb{G}_i^p| - |\mathbb{G}_i^f| \\ &= k - k(C_{0,i}) - k(C_{i,n}).\end{aligned} \tag{9.13}$$

This expression gives a relationship between the state space dimension $\rho_i(C)$ at time-i and the dimensions of the past and future subcodes, $C_{0,i}$ and $C_{i,n}$, of C with respect to time-i.

Note that $C_{0,i}$ and $C_{i,n}$ have only the all-zero codeword $\mathbb{0}$ in common. The direct-sum of $C_{0,i}$ and $C_{i,n}$, denoted by $C_{0,i} \oplus C_{i,n}$, is a subcode of C with dimension $k(C_{0,i}) + k(C_{i,n})$. The partition $C/(C_{0,i} \oplus C_{i,n})$ consists of

$$\begin{aligned}|C/(C_{0,i} \oplus C_{i,n})| &= 2^{k-k(C_{0,i})-k(C_{i,n})} \\ &= 2^{\rho_i}\end{aligned} \tag{9.14}$$

cosets of $C_{0,i} \oplus C_{i,n}$. Equation (9.14) says that the number of states in the state space $\Sigma_i(C)$ at time-i is equal to the number of cosets in the partition $C/(C_{0,i} \oplus C_{i,n})$.

Let S_i denote the subspace of C that is spanned by the rows in the submatrix $\mathbb{G}_i^s$. Then, each codeword in S_i is given by

$$\begin{aligned}\mathbb{v} &= (a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}) \cdot \mathbb{G}_i^s \\ &= a_1^{(i)} \cdot \mathbb{g}_1^{(i)} + a_2^{(i)} \cdot \mathbb{g}_2^{(i)} + \cdots + a_{\rho_i}^{(i)} \cdot \mathbb{g}_{\rho_i}^{(i)}\end{aligned} \tag{9.15}$$

where $a_l^{(i)} \in A_i^s$ for $1 \le l \le \rho_i$. It follows from the definitions of $\mathbb{G}_i^p$, $\mathbb{G}_i^f$, and $\mathbb{G}_i^s$ that

$$C = S_i \oplus (C_{0,i} \oplus C_{i,n}).$$

The 2^{ρ_i} codewords in S_i can be used as the representatives for the cosets in the partition $C/(C_{0,i} \oplus C_{i,n})$. Therefore, S_i is the coset representative space for the partition $C/(C_{0,i} \oplus C_{i,n})$. From (9.15) we see that there is *one-to-one correspondence* between $\mathbb{v}$ and the state $s_i \in \Sigma_i(C)$ defined by $(a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)})$. Because there is a one-to-one correspondence between $\mathbb{v}$ and a coset in $C/(C_{0,i} \oplus C_{i,n})$, there is a one-to-one correspondence between a state in the state space $\Sigma_i(C)$ and a coset in the partition $C/(C_{0,i} \oplus C_{i,n})$.

With the foregoing one-to-one correspondence in the trellis T, the codeword $\mathbb{v}$ given by (9.15) is represented by a path that passes through the state s_i defined by the information bits $a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}$ (i.e., a path that connects the initial state s_0 to the final state s_f through the state s_i). If we fix the information bits $a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}$ and allow the other $k - \rho_i$ information bits to vary, we obtain $2^{k-\rho_i}$ codewords of C in the coset

$$\mathbb{v} \oplus (C_{0,i} \oplus C_{i,n}) \triangleq \{\mathbb{v} + \mathbb{u} : \mathbb{u} \in C_{0,i} \oplus C_{i,n}\} \tag{9.16}$$

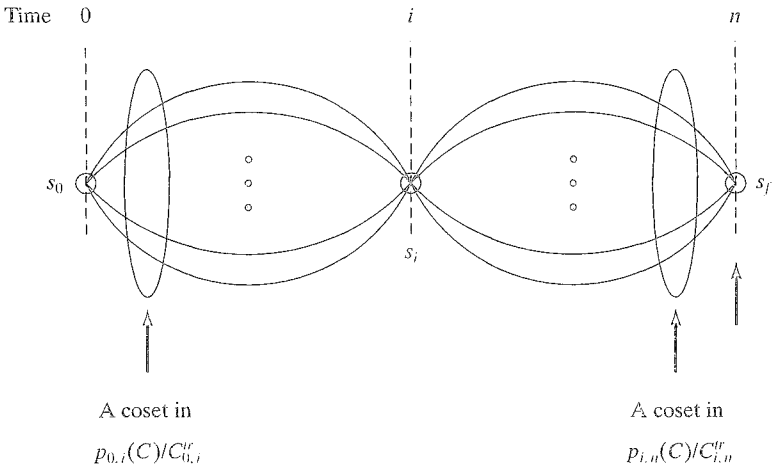

FIGURE 9.8: The paths in the code trellis that represent the $2^{k-\rho_i}$ codewords in $\mathbb{v} \oplus (C_{0,i} \oplus C_{i,n})$.

with $\mathbb{v}$ as the coset representative. In the trellis, these $2^{k-\rho_i}$ codewords are represented by the paths that connect the initial state s_0 to the final state s_f through the state s_i at time-i defined by the information bits $a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}$ as shown in Figure 9.8. Note that

$$k - \rho_i = k(C_{0,i}) + k(C_{i,n}), \tag{9.17}$$

which is simply the dimension of $C_{0,i} \oplus C_{i,n}$.

For $0 \leq i < j < n$, let $p_{i,j}(C)$ denote the linear code of length $j - i$ obtained from C by removing the first i and last $n - j$ components of each codeword in C. Every codeword in $p_{i,j}(C)$ is of the form

$$(v_i, v_{i+1}, \ldots, v_{j-1}).$$

This code is called a *punctured* (or *truncated*) code of C. Let $C^{tr}_{i,j}$ denote the punctured code of the subcode $C_{i,j}$; that is,

$$C^{tr}_{i,j} \triangleq p_{i,j}(C_{i,j}). \tag{9.18}$$

It follows from the structure of the TOGM $\mathbb{G}_{TOGM}$ that

$$k(p_{i,j}(C)) = k - k(C_{0,i}) - k(C_{j,n}) \tag{9.19}$$

and

$$k(C^{tr}_{i,j}) = k(C_{i,j}). \tag{9.20}$$

Consider the punctured code $p_{0,i}(C)$. From (9.19) we find that

$$k(p_{0,i}(C)) = k - k(C_{i,n}). \tag{9.21}$$

This punctured code contains $C_{0,i}^{tr}$ as a linear subcode with dimension $k(C_{0,i}^{tr}) = k(C_{0,i})$. We partition $p_{0,i}(C)$ based on $C_{0,i}^{tr}$. Then, the partition $p_{0,i}(C)/C_{0,i}^{tr}$ consists of

$$2^{k-k(C_{0,i})-k(C_{i,n})} = 2^{\rho_i}$$

cosets of $C_{0,i}^{tr}$. We can readily see that there is a one-to-one correspondence between the cosets in $p_{0,i}(C)/C_{0,i}^{tr}$ and the cosets in $C/(C_{0,i} \oplus C_{i,n})$, and hence a one-to-one correspondence between the cosets in $p_{0,i}(C)/C_{0,i}^{tr}$ and the states in $\Sigma_i(C)$. The codewords in a coset in $p_{0,i}(C)/C_{0,i}^{tr}$ are simply the prefixes of the codewords in their corresponding coset in $C/(C_{0,i} \oplus C_{i,n})$. Hence, the codewords in a coset of $p_{0,i}(C)/C_{0,i}^{tr}$ will take the encoder $E(C)$ to a unique state $s_i \in \Sigma_i(C)$. In the trellis T, they are the paths connecting the initial state s_0 to the state s_i as shown in Figure 9.8. Let $L(s_0, s_i)$ denote the paths in the trellis T that connect the initial state s_0 to the state s_i in $\Sigma_i(C)$. Then, $L(s_0, s_i)$ is a coset in the partition $p_{0,i}(C)/C_{0,i}^{tr}$.

Now, we consider the punctured code $p_{i,n}(C)$. The dimension of this code is

$$k(p_{i,n}(C)) = k - k(C_{0,i}). \tag{9.22}$$

It contains $C_{i,n}^{tr}$ as a linear subcode with dimension $k(C_{i,n}^{tr}) = k(C_{i,n})$. We partition $p_{i,n}(C)$ based on $C_{i,n}^{tr}$. The partition $p_{i,n}(C)/C_{i,n}^{tr}$ consists of

$$2^{k-k(C_{0,i})-k(C_{i,n})} = 2^{\rho_i}$$

cosets of $C_{i,n}^{tr}$. Again, there is a one-to-one correspondence between the cosets in $p_{i,n}(C)/C_{i,n}^{tr}$ and the cosets in $C/(C_{0,i} \oplus C_{i,n})$, and hence a one-to-one correspondence between the cosets in $p_{i,n}(C)/C_{i,n}^{tr}$ and the states in the state space $\Sigma_i(C)$. In the trellis T, the codewords in a coset $p_{i,n}(C)/C_{i,n}^{tr}$ form the paths that connect a state $s_i \in \Sigma_i(C)$ to the final state s_f as shown in Figure 9.8. Let $L(s_i, s_f)$ denote the paths in the trellis T that connect the state $s_i \in \Sigma_i(C)$ to the final state s_f. Then, $L(s_i, s_f)$ is a coset in the partition $p_{i,n}(C)/C_{i,n}^{tr}$.

For $0 \leq i < j \leq n$, let $s_i^{(\mathbf{0})}$ and $s_j^{(\mathbf{0})}$ denote two states on the all-zero path $\mathbf{0}$ in the trellis T at time-i and time-j, respectively, as shown in Figure 9.9. Let $L(s_i^{(\mathbf{0})}, s_j^{(\mathbf{0})})$ denote the paths $(v_i, v_{i+1}, \ldots, v_{j-1})$ of length $j - i$ in T that connect $s_i^{(\mathbf{0})}$ to $s_j^{(\mathbf{0})}$. Consider the paths in T that start from the initial state s_0, follow the all-zero

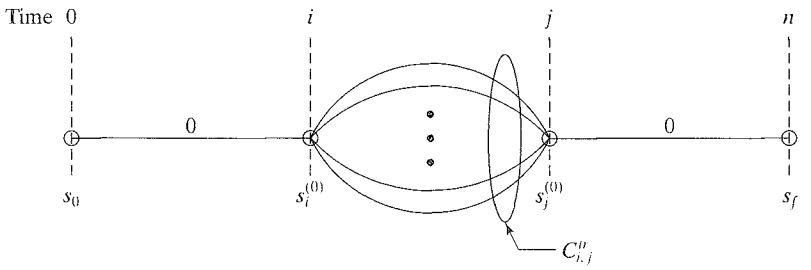

FIGURE 9.9: Paths in the code trellis that represent the codewords in $C_{i,j}$.

path $\mathbb{0}$ to the state $s_i^{(\mathbb{0})}$, pass through the paths in $L(s_i^{(\mathbb{0})}, s_j^{(\mathbb{0})})$ to the state $s_j^{(\mathbb{0})}$, then follow the all-zero path $\mathbb{0}$ until they reach the final state s_f as shown in Figure 9.9. These paths represent the codewords in the subcode $C_{i,j}$ of C, which implies that

$$L(s_i^{(\mathbb{0})}, s_j^{(\mathbb{0})}) = C_{i,j}^{tr}. \tag{9.23}$$

Let $\mathbb{v} = (v_0, v_1, \ldots, v_{n-1})$ be a path in the code trellis T. For $0 \le i < j \le n$, let $s_i^{(\mathbb{v})}$ and $s_j^{(\mathbb{v})}$ be two states on the path $\mathbb{v}$ at time-i and time-j, respectively. Let $L(s_i^{(\mathbb{v})}, s_j^{(\mathbb{v})})$ denote the paths of length $j - i$ that connect $s_i^{(\mathbb{v})}$ to $s_j^{(\mathbb{v})}$. Consider the paths in T that start from the initial state s_0, follow the path $\mathbb{v}$ to the state $s_i^{(\mathbb{v})}$, pass through the paths in $L(s_i^{(\mathbb{v})}, s_j^{(\mathbb{v})})$, then follow the path $\mathbb{v}$ until they reach the final state s_f as shown in Figure 9.10. These paths represent the codewords in the following coset of $C_{i,j}$:

$$\mathbb{v} \oplus C_{i,j} \triangleq \{\mathbb{v} + \mathbb{u} : \mathbb{u} \in C_{i,j}\}. \tag{9.24}$$

This is a coset in the partition $C/C_{i,j}$. This implies that $L(s_i^{(\mathbb{v})}, s_j^{(\mathbb{v})})$ is a coset of $C_{i,j}^{tr}$ in $p_{i,j}(C)$; that is,

$$L(s_i^{(\mathbb{v})}, s_j^{(\mathbb{v})}) = p_{i,j}(\mathbb{v}) + C_{i,j}^{tr} \in p_{i,j}(C)/C_{i,j}^{tr}, \tag{9.25}$$

where $p_{i,j}(\mathbb{v}) = (v_i, v_{i+1}, \ldots, v_{j-1})$. For any two connected states $s_i \in \Sigma_i(C)$ and $s_j \in \Sigma_j(C)$ with $0 \le i < j \le n$, they must be on a path in the trellis T. It follows from (9.25) that

$$L(s_i, s_j) \in p_{i,j}(C)/C_{i,j}^{tr}. \tag{9.26}$$

Therefore, the number of paths that connect a state $s_i \in \Sigma_i(C)$ to a state $s_j \in \Sigma_j(C)$ is given by

$$|L(s_i, s_j)| = \begin{cases} 2^{k(C_{i,j}^{tr})}, & \text{if } s_i \text{ and } s_j \text{ are connected,} \\ 0, & \text{otherwise.} \end{cases} \tag{9.27}$$

For $0 \le i < j < k \le n$, let $\Sigma_j(s_i, s_k)$ denote the set of states in $\Sigma_j(C)$ through which the paths in $L(s_i, s_k)$ connect the state s_i to the state s_k as shown in Figure 9.11. Let

$$L(s_i, s_j) \circ L(s_j, s_k) \triangleq \{\mathbb{u} \circ \mathbb{v} : \mathbb{u} \in L(s_i, s_j), \mathbb{v} \in L(s_j, s_k)\}. \tag{9.28}$$

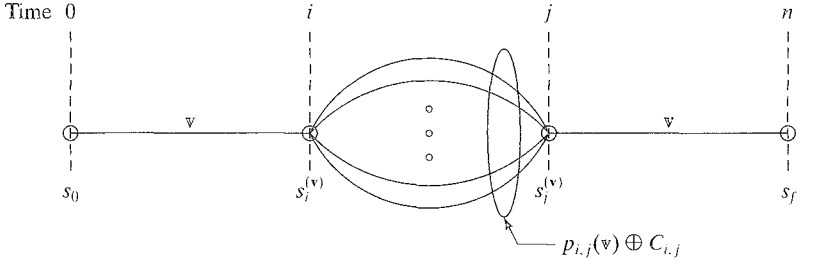

FIGURE 9.10: Paths in the code trellis that represent the codewords in the coset $\mathbb{v} \oplus C_{i,j}$.

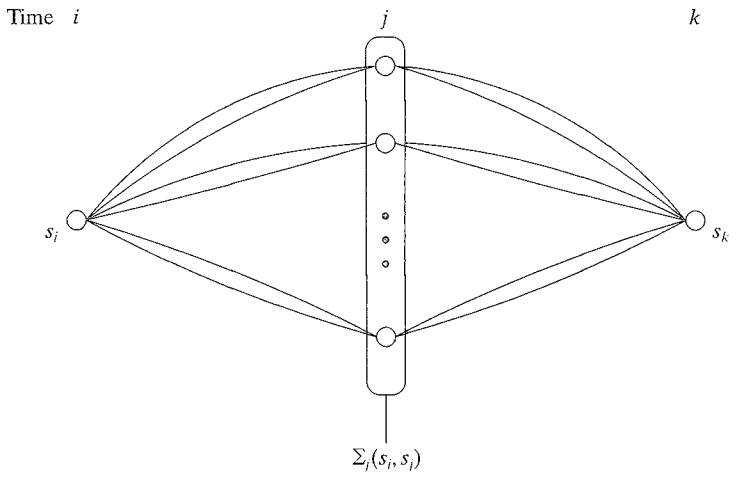

FIGURE 9.11: Connectivity between two states in the code trellis.

where $\mathbf{u} \circ \mathbf{v}$ denotes the concatenation of two sequences $\mathbf{u}$ and $\mathbf{v}$. In the trellis, $L(s_i, s_j) \circ L(s_j, s_k)$ consists of those paths in $L(s_i, s_k)$ that connect the state s_i to the state s_k through the state s_j. Then, $L(s_i, s_k)$ is the following union:

$$L(s_i, s_k) = \bigcup_{s_j \in \Sigma_j(s_i, s_k)} L(s_i, s_j) \circ L(s_j, s_k). \tag{9.29}$$

This expression displays the connectivity between two states in a code trellis.

9.5 STATE LABELING AND TRELLIS CONSTRUCTION BASED ON THE PARITY-CHECK MATRIX

Consider a binary (n, k) linear block code C with a parity-check matrix

$$\mathbb{H} = [\mathbb{h}_0, \mathbb{h}_1, \ldots, \mathbb{h}_j, \ldots, \mathbb{h}_{n-1}], \tag{9.30}$$

where, for $0 \leq j < n$, $\mathbb{h}_j$ denotes the jth column of $\mathbb{H}$ and is a binary $(n-k)$-tuple. A binary n-tuple $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ is a codeword in C if and only if

$$\mathbf{v} \cdot \mathbb{H}^T = \underbrace{(0, 0, \ldots, 0)}_{n-k}. \tag{9.31}$$

Let $\mathbb{0}_{n-k}$ denote the all-zero $(n-k)$-tuple $(0, 0, \ldots, 0)$. For $0 \leq i < n$, let $\mathbb{H}_i$ denote the submatrix that consists of the first i columns of $\mathbb{H}$; that is,

$$\mathbb{H}_i = [\mathbb{h}_0, \mathbb{h}_1, \ldots, \mathbb{h}_{i-1}]. \tag{9.32}$$

It is clear that the rank of $\mathbb{H}_i$ is at most $n-k$; that is,

$$\text{Rank}(\mathbb{H}_i) \leq n - k. \tag{9.33}$$

For each codeword $\mathbb{c} \in C_{0,i}^{tr}$,

$$\mathbb{c} \cdot \mathbb{H}_i^T = \mathbb{0}_{n-k}. \tag{9.34}$$

Therefore, $C_{0,i}^{tr}$ is the *null space* of $\mathbb{H}_i$.

Consider the partition $p_{0,i}(C)/C_{0,i}^{tr}$. Let B be a coset in $p_{0,i}/C_{0,i}^{tr}$, and $B \neq C_{0,i}^{tr}$. For every vector $\mathbb{a} \in B$,

$$\mathbb{a} \cdot \mathbb{H}_i^T \neq \mathbb{0}_{n-k}$$

and is the same for all the vectors in B; that is, for $\mathbb{a}_1, \mathbb{a}_2 \in B$ and $\mathbb{a}_1 \neq \mathbb{a}_2$,

$$\mathbb{a}_1 \cdot \mathbb{H}_i^T = \mathbb{a}_2 \cdot \mathbb{H}_i^T. \tag{9.35}$$

The $(n-k)$-tuple $\mathbb{a} \cdot \mathbb{H}_i^T$ is called the *label* for the coset B. Let B_1 and B_2 be two different cosets in $p_{0,i}(C)/C_{0,i}^{tr}$. Let $\mathbb{a}_1 \in B_1$ and $\mathbb{a}_2 \in B_2$. It follows from the theory of linear block codes (see Chapter 3) that $\mathbb{a}_1 \neq \mathbb{a}_2$, and

$$\mathbb{a}_1 \cdot \mathbb{H}_i^T \neq \mathbb{a}_2 \cdot \mathbb{H}_i^T.$$

This expression says that different cosets in $p_{0,i}(C)/C_{0,i}^{tr}$ have different labels.

Recall the mathematical formulation of the state spaces of a code trellis. There is a one-to-one correspondence between a state s_i in the state space $\Sigma_i(C)$ at time-i and a coset $B \in p_{0,i}(C)/C_{0,i}^{tr}$, and the codewords of $p_{0,i}(C)$ in B form the paths that connect the initial state s_0 to the state s_i. This one-to-one correspondence leads to another definition of a state label.

Let $L(s_0, s_i)$ denote the set of paths in the code trellis T for C that connect the initial state s_0 to a state s_i in the state space $\Sigma_i(C)$ at time-i.

> DEFINITION 9.2 For $0 \leq i \leq n$, the label of a state $s_i \in \Sigma_i(C)$ based on a parity-check matrix $\mathbb{H}$ of C, denoted by $l(s_i)$, is defined as the binary $(n-k)$-tuple
>
> $$l(s_i) \triangleq \mathbb{a} \cdot \mathbb{H}_i^T, \tag{9.36}$$
>
> for any $\mathbb{a} \in L(s_0, s_i)$. For $i = 0$, $\mathbb{H}_i = \phi$, and the initial state s_0 is labeled with the all-zero $(n-k)$-tuple, $\mathbb{0}_{n-k}$. For $i = n$, $L(s_0, s_f) = C$, and the final state s_f is also labeled with $\mathbb{0}_{n-k}$.

It follows from the preceding definition of a state label, the one-to-one correspondence between the states in $\Sigma_i(C)$ and the cosets in $p_{0,i}(C)/C_{0,i}^{tr}$ for $0 \leq i \leq n$, and (9.36) that every state $s_i \in \Sigma_i(C)$ has a unique label, and different states have different labels.

It follows from (9.33) and the foregoing definition of a state label that the number of distinct state labels at any time instant i is less than or at most equal to 2^{n-k}. Due to the uniqueness of the label for each state at any time-i, the number of states in the state space $\Sigma_i(C)$ at time-i must be upper bounded by 2^{n-k}; that is,

$$|\Sigma_i(C)| \leq 2^{n-k}$$

for $0 \leq i \leq n$. Therefore,

$$\rho_i(C) = \log_2 |\Sigma_i(C)| \leq n - k,$$

for $0 \leq i \leq n$. This expression implies that the maximum state space dimension $\rho_{\max}(C)$ defined by (9.11) satisfies the following bound:

$$\rho_{\max}(C) \leq n - k. \tag{9.37}$$

For $0 \leq i < n$, let s_i and s_{i+1} be two adjacent states with $s_i \in \Sigma_i(C)$ and $s_{i+1} \in \Sigma_{i+1}(C)$. Let v_i be the label of the branch in the code trellis T that connects the state s_i to the state s_{i+1}. The label v_i is simply the encoder output bit in the interval from time-i to time-$(i+1)$ and is given by (9.7) or (9.8). For every path $(v_0, v_1, \ldots, v_{i-1}) \in L(s_0, s_i)$, the path $(v_0, v_1, \ldots, v_{i-1}, v_i)$ obtained by concatenating $(v_0, v_1, \ldots, v_{i-1})$ with the branch v_i is a path that connects the initial state s_0 to the state s_{i+1} through the state s_i. Hence, $(v_0, v_1, \ldots, v_{i-1}, v_i) \in L(s_0, s_{i+1})$. Then, it follows from the preceding definition of a state label that

$$\begin{aligned} l(s_{i+1}) &= (v_0, v_1, \ldots, v_{i-1}, v_i) \cdot \mathbb{H}_{i+1}^T \\ &= (v_0, v_1, \ldots, v_{i-1}) \cdot \mathbb{H}_i^T + v_i \cdot \mathbb{h}_i^T \\ &= l(s_i) + v_i \cdot \mathbb{h}_i^T. \end{aligned} \tag{9.38}$$

The foregoing expression simply says that given the label of the starting state s_i, and the output code bit v_i during the interval between time-i and time-$(i+1)$, the label of the destination state s_{i+1} is uniquely determined.

Now, we present a procedure for constructing the n-section bit-level trellis diagram for a binary (n, k) linear block code C by state labeling using the parity-check matrix of the code. Let $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})$ be a binary n-tuple. For $0 \leq i \leq n$, let $p_{0,i}(\mathbf{v})$ denote the prefix of $\mathbf{v}$ that consists of the first i components; that is,

$$p_{0,i}(\mathbf{v}) = (v_0, v_1, \ldots, v_{i-1}).$$

For $i = 0$, $p_{0,0}(\mathbf{v})$ is defined as the null sequence. Suppose that the trellis has been completed up to the ith section (time-i). At this point, the rows of the TOGM $\mathbb{G}_{TOGM}$ in the set $\mathbb{G}_i^s = \{\mathbb{g}_1^{(i)}, \mathbb{g}_2^{(i)}, \ldots, \mathbb{g}_{\rho_i}^{(i)}\}$ and their corresponding information bits $a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}$ uniquely define the state space $\Sigma_i(C)$. Let

$$\mathbf{v} = a_1^{(i)} \cdot \mathbb{g}_1^{(i)} + a_2^{(i)} \cdot \mathbb{g}_2^{(i)} + \cdots + a_{\rho_i}^{(i)} \cdot \mathbb{g}_{\rho_i}^{(i)}.$$

Then, $p_{0,i}(\mathbf{v})$ is a path connecting the initial state s_0 to the state s_i defined by $a_1^{(i)}, a_2^{(i)}, \ldots, a_{\rho_i}^{(i)}$. The label of state s_i is given by

$$l(s_i) = p_{0,i}(\mathbf{v}) \cdot \mathbb{H}_i^T.$$

The $(i+1)$-section of the code trellis is constructed by taking the following four steps:

1. Identify the special row $\mathbb{g}^*$ (if any) in the submatrix $\mathbb{G}_i^f$ and its corresponding information bit a^*. Identify the special row $\mathbb{g}^0$ (if any) in the submatrix $\mathbb{G}_i^s$. Form the submatrix $\mathbb{G}_{i+1}^s$ by including $\mathbb{g}^*$ in $\mathbb{G}_i^s$ and excluding $\mathbb{g}^0$ from $\mathbb{G}_i^s$.
2. Determine the set of information bits $A_{i+1}^s = \{a_1^{(i+1)}, a_2^{(i+1)}, \ldots, a_{\rho_{i+1}}^{(i+1)}\}$ that correspond to the rows in $\mathbb{G}_{i+1}^s$. Define and label the states in $\Sigma_{i+1}(C)$.

3. For each state $s_i \in \Sigma_i(C)$, form the next output v_i code bit from either (9.7) (if there is such a row $\mathbb{g}^*$ in $\mathbb{G}_i^f$ at time-i) or (9.8) (if there is no such row $\mathbb{g}^*$ in $\mathbb{G}_i^f$ at time-i).
4. For each possible value of v_i (two if computed from (9.7) and one if computed from (9.8)), connect the state s_i to the state $s_{i+1} \in \Sigma_{i+1}(C)$ with label

$$l(s_{i+1}) = l(s_i) + v_i \cdot \mathbb{h}_i^T.$$

Label the connecting branch, denoted by $L(s_i, s_{i+1})$, with v_i. This completes the construction of the $(i+1)$th section of the trellis.

Repeat the preceding steps until the entire code trellis is constructed.

EXAMPLE 9.11

The (8, 4) RM code given in Example 9.1 is a self-dual code. Therefore, a generator matrix is also a parity-check matrix. Suppose we choose the parity-check matrix as follows:

$$\mathbb{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

Using this parity-check matrix for state labeling and following the foregoing trellis construction steps, we obtain the 8-section trellis with state labels shown in Figure 9.12. To illustrate the construction process, we assume that the trellis has been completed up to time-3. At this instant, $\mathbb{G}_3^s = \{\mathbb{g}_0, \mathbb{g}_1, \mathbb{g}_2\}$ and $A_3^s = \{a_0, a_1, a_2\}$ are known. The eight states in $\Sigma_3(C)$ are defined by the eight combinations of a_0, a_1, and a_2. These eight states and their labels are given in Table 9.5.

Now, we want to construct the fourth section of the trellis up to time-4. At time-3, from the TOGM $\mathbb{G}_{TOGM}$ (see Example 9.1), we find that $\mathbb{g}^0 = \mathbb{g}_0$ and there is no such row $\mathbb{g}^*$ with leading 1 at the fourth bit position. Therefore, $\mathbb{G}_4^s = \{\mathbb{g}_1, \mathbb{g}_2\}$, and $A_4^s = \{a_1, a_2\}$. The four states in $\Sigma_4(C)$ at time-4 are defined by the four

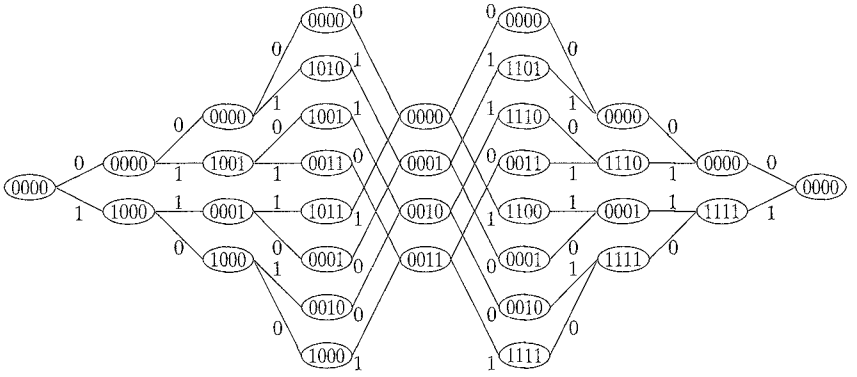

FIGURE 9.12: An 8-section trellis for the (8, 4) RM code with state labeling by the parity-check matrix.

TABLE 9.5: Labels of the states at time-3 for the (8, 4) RM code based on the parity-check matrix.

	States Defined by (a_0, a_1, a_2)	State Label
$s_3^{(0)}$	(000)	(0000)
$s_3^{(1)}$	(001)	(1010)
$s_3^{(2)}$	(010)	(1001)
$s_3^{(3)}$	(011)	(0011)
$s_3^{(4)}$	(100)	(1011)
$s_3^{(5)}$	(101)	(0001)
$s_3^{(6)}$	(110)	(0010)
$s_3^{(7)}$	(111)	(1001)

TABLE 9.6: Paths defined by the four states at time-3.

(a_1, a_2)	Path
(0, 0)	$\mathbf{v}_0 = (00000000)$
(0, 1)	$\mathbf{v}_1 = (00111100)$
(1, 0)	$\mathbf{v}_2 = (01011010)$
(1, 1)	$\mathbf{v}_3 = (01100110)$

TABLE 9.7: Labels of states at time-4 for the (8, 4) RM code.

	State Defined by (a_1, a_2)	State Label
$s_4^{(0)}$	(00)	(0000)
$s_4^{(1)}$	(01)	(0001)
$s_4^{(2)}$	(10)	(0010)
$s_4^{(3)}$	(11)	(0011)

combinations of a_1 and a_2. The four codewords generated by the rows in $\mathbf{G}_4^s$ are given in Table 9.6. The four paths that connect the initial state s_0 to the four states, denoted by $s_4^{(0)}$, $s_4^{(1)}$, $s_4^{(2)}$, and $s_4^{(3)}$, in $\Sigma_4(C)$ are

$$p_{0,4}(\mathbf{v}_0) = (0000), \quad p_{0,4}(\mathbf{v}_1) = (0011),$$
$$p_{0,4}(\mathbf{v}_2) = (0101), \quad p_{0,4}(\mathbf{v}_3) = (0110).$$

The submatrix $\mathbf{H}_4$ is

$$\mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}.$$

From $p_{0,4}(\mathbf{v}_j)$, with $0 \le j \le 3$ and $\mathbf{H}_4$, we can determine the labels of the four states, $s_4^{(0)}$, $s_4^{(1)}$, $s_4^{(2)}$ and $s_4^{(3)}$, in $\Sigma_4(C)$, which are given in Table 9.7. The four states and their labels are shown in Figure 9.13 at time-4. Now, suppose the encoder is in the state $s_3^{(2)}$ with label $l(s_3^{(2)}) = (1001)$ at time-3. Because no such row $\mathbf{g}^*$ exists at time

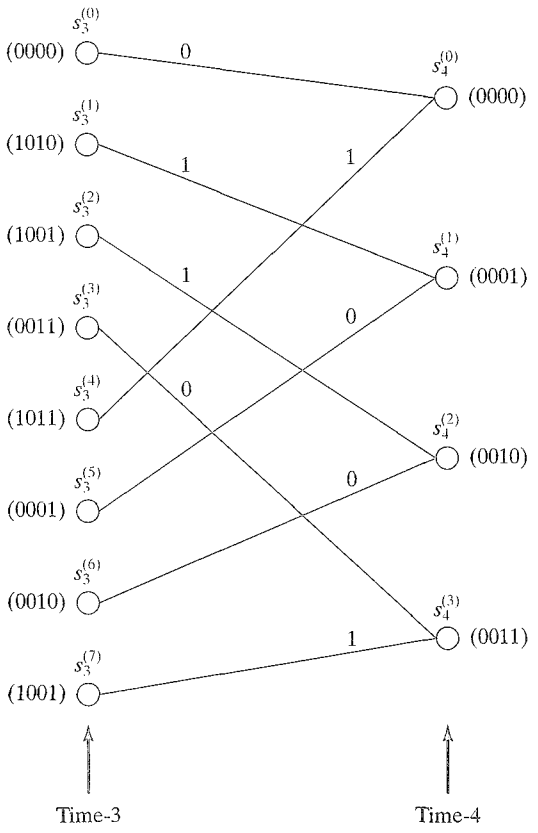

FIGURE 9.13: State labels at the two ends of the fourth section of the trellis for the (8, 4) RM code.

$i = 3$, the output code bit v_3 is computed from (9.8) as follows:

$$\begin{aligned} v_3 &= a_0 \cdot g_{03} + a_1 \cdot g_{13} + a_2 \cdot g_{23} \\ &= 0 \cdot 1 + 1 \cdot 1 + 0 \cdot 1 \\ &= 1. \end{aligned}$$

Then, the state $s_3^{(2)}$ is connected to the state in $\Sigma_4(C)$ with label

$$\begin{aligned} l(s_3^{(2)}) + v_3 \cdot \mathbb{h}_3^T &= (1001) + 1 \cdot (1011) \\ &= (0010), \end{aligned}$$

which is state $s_4^{(2)}$, as shown in Figure 9.13. The connecting branch is labeled with $v_3 = 1$. The connections from the other states in $\Sigma_3(C)$ to the states in $\Sigma_4(C)$ are made in the same manner.

State labeling based on the state-defining information sets requires k (or $\rho_{\max}(C)$) bits to label each state of the trellis; however, state labeling based on the parity-check matrix requires $n - k$ bits to label each state of the trellis. In the next section we show that $\rho_{\max}(C) \leq \min\{k, n - k\}$. Therefore, state labeling based on state-defining information sets and using $\rho_{\max}(C)$ bits to label each state in the trellis is the most economical labeling method.

EXAMPLE 9.12

A parity-check matrix of the (16, 11) RM code considered in Example 9.10 is

$$\mathbb{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

State labeling based on this matrix requires 5 bits. The 16-section trellis for the (16, 11) RM code with state labels based on $\mathbb{H}$ is shown in Figure 9.14. Labeling states based on $\mathbb{H}$ requires 1 bit more than state labeling based on the maximum state-defining information set shown in Example 9.10.

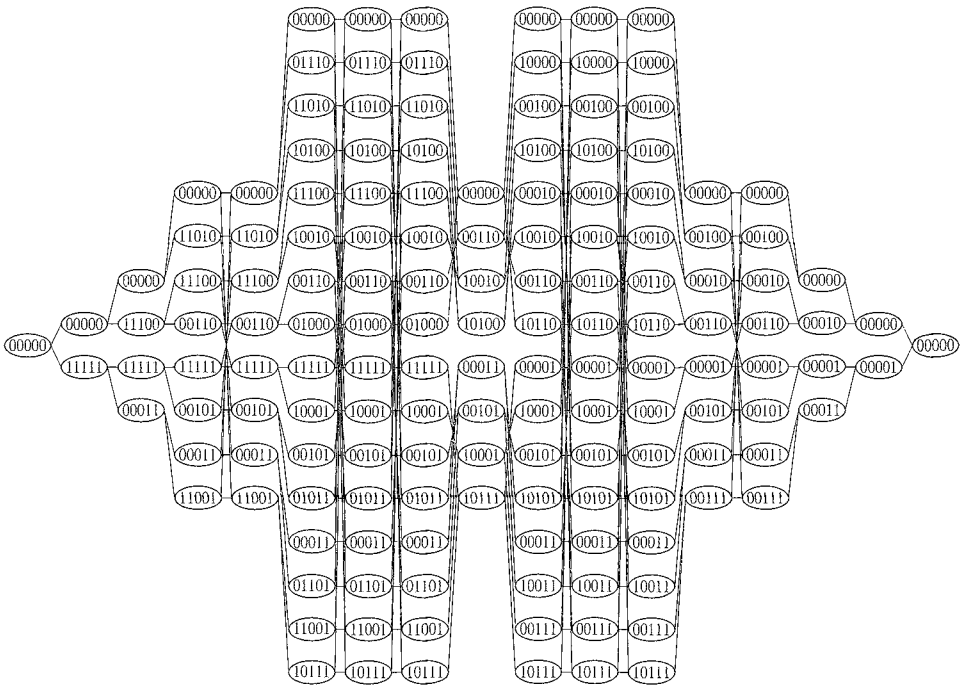

FIGURE 9.14: A 16-section trellis for the (16, 11) RM code with state labeling by parity-check matrix.

9.6 TRELLIS COMPLEXITY AND SYMMETRY

Trellis complexity is, in general, measured in terms of the state and branch complexities. These two complexities determine the storage and computation requirements of a trellis-based decoding algorithm.

For a binary (n, k) linear block code C, the state complexity of an n-section bit-level code trellis is measured by its maximum state space dimension $\rho_{\max}(C)$ defined by (9.11), which gives the size $2^{\rho_{\max}(C)}$ of the maximum state space of the code trellis T. In (9.12) and (9.37), we showed that $\rho_{\max}(C)$ is upper bounded by both k and $n-k$. This implies that $\rho_{\max}(C)$ must satisfy the following bound:

$$\rho_{\max}(C) \leq \min\{k, n-k\}. \tag{9.39}$$

This bound was first proved by Wolf [4]. In general, this bound is quite loose. For example, consider the (8, 4) RM code. For this code, $k = n - k = 4$; however, $\rho_{\max}(C) = 3$. The third-order RM code $RM(3, 6)$ of length 64 is a (64, 42) linear code. For this code, $\min\{k, n-k\} = 22$; however $\rho_{\max}(C) = 14$. We see that there is a big gap between the bound and $\rho_{\max}(C)$; however, for cyclic (or shortened cyclic) codes, the bound $\min\{k, n-k\}$ gives the exact state complexity, as will be shown later.

Next, we show that an (n, k) linear code C and its dual code have the same state complexity. Let $C^{\perp}$ denote the dual code of C. Then, $C^{\perp}$ is an $(n, n-k)$ linear block code. Consider the n-section trellis diagram for $C^{\perp}$. For $0 \leq i \leq n$, let $\Sigma_i(C^{\perp})$ denote the state space of $C^{\perp}$ at time-i. Then, there is a one-to-one correspondence between the states in $\Sigma_i(C^{\perp})$ and the cosets in the partition $p_{0,i}(C^{\perp})/C_{0,i}^{\perp,tr}$, where $C_{0,i}^{\perp,tr}$ denotes the truncation of $C_{0,i}^{\perp}$ in the interval $[0, i-1]$. Therefore, the dimension of $\Sigma_i(C^{\perp})$ is given by

$$\rho_i(C^{\perp}) = k(p_{0,i}(C^{\perp})) - k(C_{0,i}^{\perp,tr}). \tag{9.40}$$

Note that $p_{0,i}(C^{\perp})$ is the dual code of $C_{0,i}^{tr}$, and $C_{0,i}^{\perp,tr}$ is the dual code of $p_{0,i}(C)$. Therefore,

$$\begin{aligned} k(p_{0,i}(C^{\perp})) &= i - k(C_{0,i}^{tr}) \\ &= i - k(C_{0,i}) \end{aligned} \tag{9.41}$$

and

$$k(C_{0,i}^{\perp,tr}) = i - k(p_{0,i}(C)). \tag{9.42}$$

It follows from (9.40) through (9.42) that

$$\rho_i(C^{\perp}) = k(p_{0,i}(C)) - k(C_{0,i}).$$

Because $k(p_{0,i}(C)) = k - k(C_{i,n})$, we have

$$\rho_i(C^{\perp}) = k - k(C_{0,i}) - k(C_{i,n}). \tag{9.43}$$

From (9.13) and (9.43), we find that for $0 \leq i \leq n$,

$$\rho_i(C^{\perp}) = \rho_i(C). \tag{9.44}$$

This expression says that C and its dual code $C^{\perp}$ have the same state complexity. Therefore, we can analyze the state complexity of a code trellis from either the code or its dual code; however, we must note that a code and its dual code, in general, have different branch complexities, even though they have the same state complexity.

EXAMPLE 9.13

Consider the (16, 11) RM code given in Example 9.10. The dual code of this code is the first-order RM code $RM(1, 4)$ of length 16. It is a (16, 5) code with a minimum distance of 8. Its TOGM is

$$\mathbf{G}_{TOGM} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

From this matrix, we find the state space dimension profile of the code,

$$(0, 1, 2, 3, 3, 4, 4, 4, 3, 4, 4, 4, 3, 3, 2, 1, 0),$$

which is exactly the same as the state space dimension profile of the trellis of the (16, 11) RM code. The state-defining information sets and state labels are given in Table 9.8. The 16-section trellis for the code is shown in Figure 9.15. State labeling is done based on the state-defining information sets. State labeling based on the parity-check matrix of the code would require 11 bits.

TABLE 9.8: State-defining sets and labels for the 16-section trellis for the (16, 5) RM code.

i	$\mathbf{G}_i^s$	a^*	a^0	A_i^s	State Label
0	ϕ	a_0	—	ϕ	(0000)
1	$\{\mathbf{g}_0\}$	a_1	—	$\{a_0\}$	(a_0000)
2	$\{\mathbf{g}_0, \mathbf{g}_1\}$	a_2	—	$\{a_0, a_1\}$	(a_0a_100)
3	$\{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2\}$	—	—	$\{a_0, a_1, a_2\}$	$(a_0a_1a_20)$
4	$\{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2\}$	a_3	—	$\{a_0, a_1, a_2\}$	$(a_0a_1a_20)$
5	$\{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3\}$	—	—	$\{a_0, a_1, a_2, a_3\}$	$(a_0a_1a_2a_3)$
6	$\{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3\}$	—	—	$\{a_0, a_1, a_2, a_3\}$	$(a_0a_1a_2a_3)$
7	$\{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3\}$	—	a_0	$\{a_0, a_1, a_2, a_3\}$	$(a_0a_1a_2a_3)$
8	$\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3\}$	a_4	—	$\{a_1, a_2, a_3\}$	$(a_1a_2a_30)$
9	$\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3, \mathbf{g}_4\}$	—	—	$\{a_1, a_2, a_3, a_4\}$	$(a_1a_2a_3a_4)$
10	$\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3, \mathbf{g}_4\}$	—	—	$\{a_1, a_2, a_3, a_4\}$	$(a_1a_2a_3a_4)$
11	$\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3, \mathbf{g}_4\}$	—	a_3	$\{a_1, a_2, a_3, a_4\}$	$(a_1a_2a_3a_4)$
12	$\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_4\}$	—	—	$\{a_1, a_2, a_4\}$	$(a_1a_2a_40)$
13	$\{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_4\}$	—	a_2	$\{a_1, a_2, a_4\}$	$(a_1a_2a_40)$
14	$\{\mathbf{g}_1, \mathbf{g}_4\}$	—	a_1	$\{a_1, a_4\}$	(a_1a_400)
15	$\{\mathbf{g}_4\}$	—	a_4	$\{a_4\}$	(a_4000)
16	ϕ	—	—	ϕ	(0000)

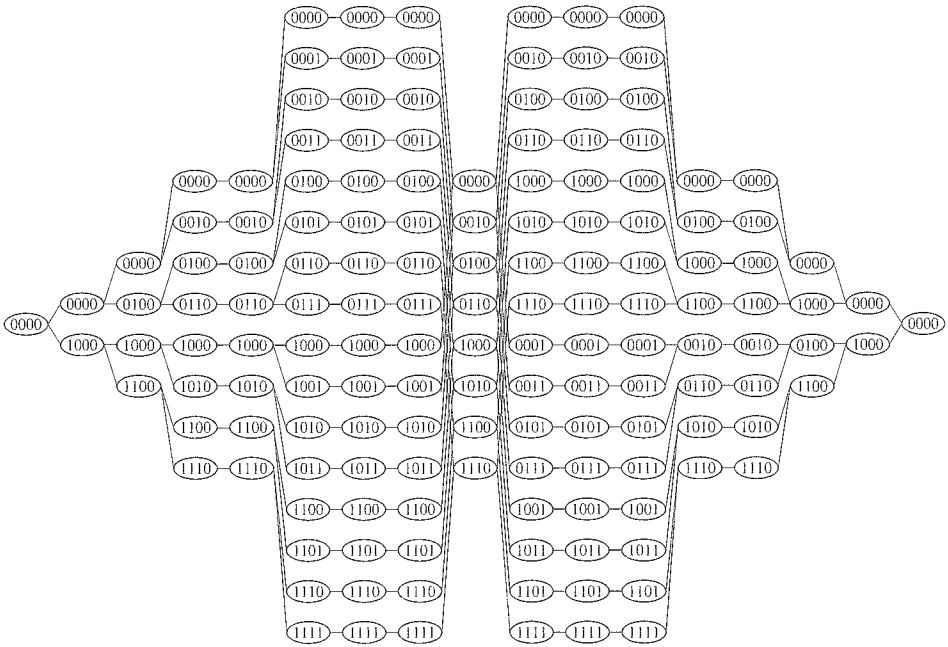

FIGURE 9.15: A 16-section trellis for the (16, 5) RM code with state labeling by the state-defining information set.

Let T be an n-section trellis for an (n, k) code C with state space dimension profile $(\rho_0, \rho_1, \ldots, \rho_n)$. The trellis T is said to be *minimal* if for any other n-section trellis T' for C with state space dimension profile $(\rho'_0, \rho'_1, \ldots, \rho'_n)$ the following inequality holds:

$$\rho_i \leq \rho'_i,$$

for $0 \leq i \leq n$. A minimal trellis is unique within isomorphism; that is, two minimal trellises for the same code are structurally identical. A minimal trellis results in a minimal total number of states in the trellis. In fact, the inverse is also true: a trellis with a minimum total number of states is a minimal trellis. The formulation of state spaces and the construction based on the trellis-oriented generator matrix of an (n, k) linear block code given in Section 9.2 result in a minimal trellis. The proof is not given here but can be found in [7].

From (9.13) we see that the state space dimension ρ_i at time-i depends on the dimensions of the past and future codes, $C_{0,i}$ and $C_{i,n}$. For a given code C, $k(C_{0,i})$ and $k(C_{i,n})$ are fixed. Given an (n, k) linear code C, a permutation of the orders of the bit (or symbol) positions results in an equivalent code C' with the same weight distribution and the same error performance; however, different permutations of bit positions may result in different dimensions, $k(C_{0,i})$ and $k(C_{i,n})$, of the past and future subcodes, $C_{0,i}$ and $C_{i,n}$, and hence different state space dimensions ρ_i at time-i. A permutation that yields the smallest state space dimension at every time of the code trellis is called an *optimum permutation* (or bit ordering). It is clear that an optimum permutation reduces the state complexity and is often desirable. Optimum

permutation is *hard to find*, however. Optimum permutations for RM codes are known [11], but they are unknown for other classes of codes. The construction of RM code given in Section 4.3 gives the minimum state complexity.

The *branch complexity* of an n-section trellis diagram for an (n, k) linear code C is defined as the total number of branches in the trellis. This complexity determines the number of computations required in a trellis-based decoding algorithm to decode a received sequence. An n-section trellis diagram for an (n, k) linear block code is said to be a *minimal branch (*or *edge) trellis diagram* if it has the smallest branch complexity. A minimal trellis diagram has the smallest branch complexity.

Consider the n-section trellis diagram T for C that is constructed based on the rules and procedures described in Section 9.2. Recall that at time-i with $0 \leq i < n$, there are two branches diverging from a state in $\Sigma_i(C)$ if there exists a row $\mathbf{g}^*$ in $\mathbf{G}_i^f$; and there is only one branch diverging from a state in $\Sigma_i(C)$ if there exists no such row $\mathbf{g}^*$ in $\mathbf{G}_i^f$. The existence of $\mathbf{g}^*$ in $\mathbf{G}_i^f$ implies that there is a current input information bit $a^* \in A_i^f$ at time-i. We define

$$I_i(a^*) \triangleq \begin{cases} 1, & \text{if } a^* \notin A_i^f, \\ 2, & \text{if } a^* \in A_i^f. \end{cases} \tag{9.45}$$

Let $\mathcal{E}$ denote the total number of branches in T. Then,

$$\begin{aligned} \mathcal{E} &= \sum_{i=0}^{n-1} |\Sigma_i(C)| \cdot I_i(a^*) \\ &= \sum_{i=0}^{n-1} 2^{\rho_i} \cdot I_i(a^*). \end{aligned} \tag{9.46}$$

For $0 \leq i < n$, $2^{\rho_i} \cdot I_i(a^*)$ is simply the number of branches in the ith section of trellis T.

EXAMPLE 9.14

Again we consider the (8, 4) RM code given in Example 9.1. From Table 9.2 we find that

$$I_0(a^*) = I_1(a^*) = I_2(a^*) = I_4(a^*) = 2$$

and

$$I_3(a^*) = I_5(a^*) = I_6(a^*) = I_7(a^*) = 1.$$

The state space dimension profile of the 8-section trellis for the code is (0, 1, 2, 3, 2, 3, 2, 1, 0). From (9.46) we have

$$\begin{aligned} \mathcal{E} &= 2^0 \cdot 2 + 2^1 \cdot 2 + 2^2 \cdot 2 + 2^3 \cdot 1 + 2^2 \cdot 2 + 2^3 \cdot 1 + 2^2 \cdot 1 + 2^1 \cdot 1 \\ &= 2 + 4 + 8 + 8 + 8 + 8 + 4 + 2 \\ &= 44. \end{aligned}$$

Cyclic codes form a large subclass of linear block codes. Many good codes are cyclic codes. The trellis structure of cyclic codes can be analyzed easily. Consider an (n, k) cyclic code C over $GF(2)$ with generator polynomial

$$g(X) = 1 + g_1 X + g_2 X^2 + \cdots + g_{n-k-1} X^{n-k-1} + X^{n-k}.$$

A generator matrix for this code is given in (5.9). For convenience, we reproduce it here

$$\mathbb{G} = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{k-1} \end{bmatrix} = \begin{bmatrix} 1 & g_1 & g_2 & \cdots & \cdots & g_{n-k-1} & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & g_1 & g_2 & \cdots & \cdots & g_{n-k-1} & 1 & 0 & \cdots & 0 \\ & & & & & \ddots & & & & & \\ 0 & 0 & \cdots & 0 & 1 & g_1 & g_2 & \cdots & g_{n-k-1} & 1 & \end{bmatrix}. \quad (9.47)$$

We readily see that the generator matrix is already in trellis-oriented form. For $0 \leq i < k$, the time span of the i-row g_i is

$$\tau(g_i) = [i, n - k + 1 + i].$$

(or the bit span $\phi(g_i) = [i, n - k + i]$). The active time spans of all the rows have the same length, $n - k$.

Now, we consider the n-section bit-level trellis for this (n, k) cyclic code. There are two cases to be considered: $k > n - k$ and $k \leq n - k$. Consider the case $k > n - k$. For $1 \leq i < n - k$, the number of rows in $\mathbb{G}$ whose active time spans contain the time index i is i. These rows are simply the first i rows. For $n - k < i \leq k$, the number of rows whose active time spans contain the time index i is $n - k$. For $k < i \leq n$, the number of rows whose active time spans contain the time index i is $n - i$. Because $i > k$, $n - i < n - k$. From the preceding analysis, we see that the maximum state space dimension is $\rho_{\max}(C) = n - k$, and the state space profile is

$$(0, 1, \ldots, n - k - 1, n - k, \ldots, n - k, n - k - 1, \ldots, 1, 0).$$

Next, we consider the second case for which $k \leq n - k$. For $1 \leq i < k$, the number of rows in $\mathbb{G}$ whose active time spans contain the time index i is i (the first i rows in $\mathbb{G}$). For $k \leq i \leq n - k$, the number of rows whose active time spans contain the time index i is k. For $n - k < i \leq n$, the number of rows in $\mathbb{G}$ whose active time spans contain i is $n - i$. Because $i > n - k$, $n - i < k$. From the foregoing analysis, we find that the maximum state space dimension is $\rho_{\max}(C) = k$, and the state space dimension profile is

$$(0, 1, \ldots, k - 1, k, \ldots, k, k - 1, \ldots, 1, 0).$$

Combining the results of the preceding two cases, we conclude that for an (n, k) cyclic code, the maximum state space dimension is

$$\rho_{\max}(C) = \min\{k, n - k\}.$$

This is to say that a code in cyclic form has the worst state complexity; that is, it meets the upper bound on the state complexity. To reduce the state complexity of a cyclic code, a proper permutation on the bit positions is needed.

EXAMPLE 9.15

Consider the $(7, 4)$ cyclic Hamming code generated by $\mathbf{g}(X) = 1 + X + X^3$. Its generator matrix in TOF is

$$\mathbb{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1\,1\,0\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0 \\ 0\,0\,1\,1\,0\,1\,0 \\ 0\,0\,0\,1\,1\,0\,1 \end{bmatrix}.$$

By examining the generator matrix, we find that the trellis state space dimension profile is $(0, 1, 2, 3, 3, 2, 1, 0)$. Therefore, $\rho_{\max}(C) = 3$. The 7-section trellis for this code is shown in Figure 9.16, the state-defining information sets and the state labels are given in Table 9.9.

Next, we derive a special symmetry structure for trellises of some linear block codes. This structure is quite useful for implementing a trellis-based decoder, especially in gaining decoding speed. Consider the TOGM of a binary (n, k) linear block code of even length n,

$$\mathbb{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix}.$$

Suppose the TOGM $\mathbb{G}$ has the following symmetry property: for each row $\mathbf{g}$ in $\mathbb{G}$ with bit span $\phi(\mathbf{g}) = [a, b]$, there exists a row $\mathbf{g}'$ in $\mathbb{G}$ with bit span $\phi(\mathbf{g}') =$

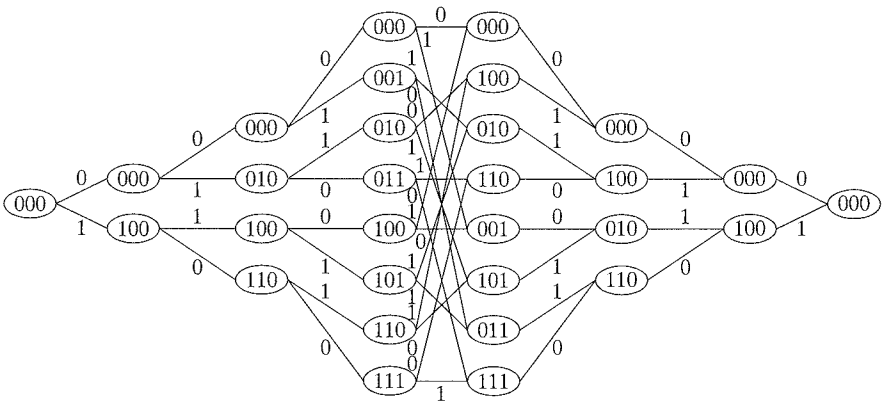

FIGURE 9.16: The 7-section trellis diagram for the (7, 4) Hamming code with 3-bit state label.

TABLE 9.9: State-defining sets and state labels for the 8-section trellis for the (8, 4) RM code.

i	$\mathbb{G}_i^s$	a^*	a^0	A_i^s	State Label
0	ϕ	a_0	—	ϕ	(000)
1	$\{\mathbb{g}_0\}$	a_1	—	$\{a_0\}$	$(a_0 00)$
2	$\{\mathbb{g}_0, \mathbb{g}_1\}$	a_2	—	$\{a_0, a_1\}$	$(a_0 a_1 0)$
3	$\{\mathbb{g}_0, \mathbb{g}_1, \mathbb{g}_2\}$	a_3	a_0	$\{a_0, a_1, a_2\}$	$(a_0 a_1 a_2)$
4	$\{\mathbb{g}_1, \mathbb{g}_2, \mathbb{g}_3\}$	—	a_1	$\{a_1, a_2, a_3\}$	$(a_1 a_2 a_3)$
5	$\{\mathbb{g}_2, \mathbb{g}_3\}$	—	a_2	$\{a_2, a_3\}$	$(a_2 a_3 0)$
6	$\{\mathbb{g}_3\}$	—	a_3	$\{a_3\}$	$(a_3 00)$
7	ϕ	—	—	ϕ	(000)

$[n-1-b, n-1-a]$. With this symmetry in $\mathbb{G}$, we can readily see that for $0 \leq i \leq n/2$, the number of rows in $\mathbb{G}$ that are active at time-$(n-i)$ is equal to the number of rows in $\mathbb{G}$ that are active at time-i. This implies that $|\Sigma_{n-i}(C)| = |\Sigma_i(C)|$ (or $\rho_{n-i} = \rho_i$) for $0 \leq i \leq n/2$. We can permute the rows of $\mathbb{G}$ such that the resultant matrix, denoted by $\mathbb{G}'$, is in a *reverse trellis-oriented form*:

1. The trailing 1 of each row appears in a column before the trailing 1 of any row below it.
2. No two rows have their leading 1's in the same column.

If we rotate the matrix $\mathbb{G}'$ 180° counterclockwise, we obtain a matrix $\mathbb{G}''$ in which the ith row $\mathbb{g}_i''$ is simply the $(k-1-i)$th row $\mathbb{g}_{k-1-i}'$ of $\mathbb{G}'$ in reverse order (the trailing 1 of $\mathbb{g}_{k-1-i}'$ becomes the leading 1 of $\mathbb{g}_i''$, and the leading 1 of $\mathbb{g}_{k-1-i}'$ becomes the trailing 1 of $\mathbb{g}_i''$). From the foregoing, we see that $\mathbb{G}''$ and $\mathbb{G}$ are structurally identical in the sense that $\phi(\mathbb{g}_i'') = \phi(\mathbb{g}_i)$ for $0 \leq i < k$. Consequently, the n-section trellis T for C has the following *mirror symmetry* [7, 37]: the last $n/2$ sections of T form the mirror image of the first $n/2$ sections of T (not including the path labels).

EXAMPLE 9.16

Consider the (8, 4) RM code given in Example 9.1 with TOGM

$$\mathbb{G}_{TOGM} = \begin{bmatrix} \mathbb{g}_0 \\ \mathbb{g}_1 \\ \mathbb{g}_2 \\ \mathbb{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Examining the rows of $\mathbb{G}_{TOGM}$, we find that $\phi(\mathbb{g}_0) = [0, 3]$, $\phi(\mathbb{g}_3) = [4, 7]$, and $\mathbb{g}_0$ and $\mathbb{g}_3$ are symmetrical with each other. Row $\mathbb{g}_1$ has bit span $[1, 6]$ and is symmetrical with itself. Row $\mathbb{g}_2$ has bit span $[2, 5]$ and is also symmetrical with itself. Suppose we permute $\mathbb{g}_1$ and $\mathbb{g}_2$. We obtain the following matrix in reverse trellis-oriented form:

$$\mathbb{G}' = \begin{bmatrix} \mathbb{g}_0' \\ \mathbb{g}_1' \\ \mathbb{g}_2' \\ \mathbb{g}_3' \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Rotating $\mathbb{G}'$ 180° counterclockwise, we obtain the following matrix:

$$\mathbb{G}'' = \begin{bmatrix} \mathbb{g}_0'' \\ \mathbb{g}_1'' \\ \mathbb{g}_2'' \\ \mathbb{g}_3'' \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

We find that $\mathbb{G}''$ and $\mathbb{G}$ are in fact identical, not just structurally identical. Therefore, the 8-section trellis T for the (8, 4) RM code has mirror symmetry with respect to boundary location 4: the last four sections form the mirror image of the first four sections, as shown in Figures 9.3 and 9.6.

Figure 9.7 shows that the 16-section bit-level trellis for the (16, 11) RM code also has mirror-image symmetry. In fact, trellises for all RM codes have mirror-image symmetry.

For the case in which n is odd, if the TOGM $\mathbb{G}_{TOGM}$ of a binary (n, k) code C has the mirror symmetry property, then the last $(n-1)/2$ sections of the n-section trellis T for C form the mirror image of the first $(n-1)/2$ sections of T.

From (9.47) we readily see that the generator matrix of a cyclic code has the mirror symmetry property. Therefore, the trellises of all cyclic codes have mirror-image symmetry. Figure 9.16 displays the mirror-image symmetry of the (7,4) Hamming code.

9.7 TRELLIS SECTIONALIZATION AND PARALLEL DECOMPOSITION

In a bit-level trellis diagram, every time instant in the encoding interval $\Gamma = \{0, 1, 2, \ldots, n\}$ is a section boundary location, and every branch represents a code bit. It is possible to sectionalize a bit-level trellis with section boundary locations at selected instants in Γ. This sectionalization results in a trellis in which a branch may represent multiple code bits, and two adjacent states may be connected by multiple branches. Proper sectionalization may result in useful trellis structural properties and allow us to devise efficient trellis-based decoding algorithms or to simplify decoder implementation.

For a positive integer $\nu \leq n$, let

$$\Lambda \triangleq \{t_0, t_1, t_2, \ldots, t_\nu\}$$

be a subset of $\nu + 1$ time instants in the encoding interval $\Gamma = \{0, 1, 2, \ldots, n\}$ for an (n, k) linear block code C with $0 = t_0 < t_1 < t_2 < \cdots < t_\nu = n$. A ν-section trellis diagram for C with section boundaries at the locations (time instants) in Λ, denoted by $T(\Lambda)$, can be obtained from the n-section trellis T by (1) deleting every state in $\Sigma_t(C)$ for $t \in \{0, 1, \ldots, n\} \setminus \Lambda$ and every branch entering or leaving a deleted state, and (2) for $1 \leq j \leq \nu$, connecting a state $s \in \Sigma_{t_{j-1}}$ to a state $s' \in \Sigma_{t_j}$ by a branch with label α if and only if there is a path with label α from state s to state s' in the n-section trellis T. In this ν-section trellis, a branch connecting a state $s \in \Sigma_{t_{j-1}}$ to a state $s' \in \Sigma_{t_j}$ represents $(t_j - t_{j-1})$ code symbols. The state space $\Sigma_{t_{j-1}}(C)$ at time-t_{j-1}, the state space $\Sigma_{t_j}(C)$ at time-t_j, and all the branches between states in $\Sigma_{t_{j-1}}(C)$ and states in $\Sigma_{t_j}(C)$, form the jth section of $T(\Lambda)$. The length of this trellis

section is $t_j - t_{j-1}$. If the lengths of all the sections are the same, $T(\Lambda)$ is said to be *uniformly sectionalized*. The state space dimension profile for this sectionalized trellis is an $(\nu + 1)$-tuple,

$$(\rho_0, \rho_{t_1}, \rho_{t_2}, \ldots, \rho_{t_{\nu-1}}, \rho_n),$$

where

$$\begin{aligned} \rho_{t_j} &= \log_2 |\Sigma_{t_j}(C)| \\ &= k - k(C_{0,t_j}) - k(C_{t_j,n}). \end{aligned} \tag{9.48}$$

The maximum state space dimension of the ν-section trellis $T(\Lambda)$ is given by

$$\rho_{\nu,\max}(C) \triangleq \max_{0 \le j \le \nu} \rho_{t_j}. \tag{9.49}$$

If the section boundary locations $t_0, t_1, \ldots, t_\nu$ are chosen at the places where $\rho_{t_1}, \rho_{t_2}, \ldots, \rho_{t_{\nu-1}}$ are small, then the resultant ν-section code trellis $T(\Lambda)$ has a small state space complexity; however, sectionalization, in general, results in an increase in branch complexity. Hence, it is important to properly choose the section boundary locations to provide a good trade-off between state and branch complexities.

EXAMPLE 9.17

Consider the 8-section bit-level trellis T for the (8, 4) RM code shown in Figure 9.6. Suppose we choose $\nu = 4$ and the section boundary set $\Lambda = \{0, 2, 4, 6, 8\}$. Following the foregoing rules of sectionalization, we obtain a uniform 4-section trellis diagram as shown in Figure 9.17, in which every branch represents 2 code bits. The state space dimension profile for this 4-section trellis is (0, 2, 2, 2, 0), and the maximum state space dimension is $\rho_{4,\max}(C) = 2$. The trellis still possesses mirror-image symmetry. Furthermore, the sectionalization decomposes the trellis into two parallel and structurally identical (or isomorphic) subtrellises without cross connections between them.

EXAMPLE 9.18

Consider the 16-section trellis for the (16, 11) RM code shown in Figure 9.7. Suppose we choose $\nu = 4$ and the section boundary set $\Lambda = \{0, 4, 8, 12, 16\}$. The result is a

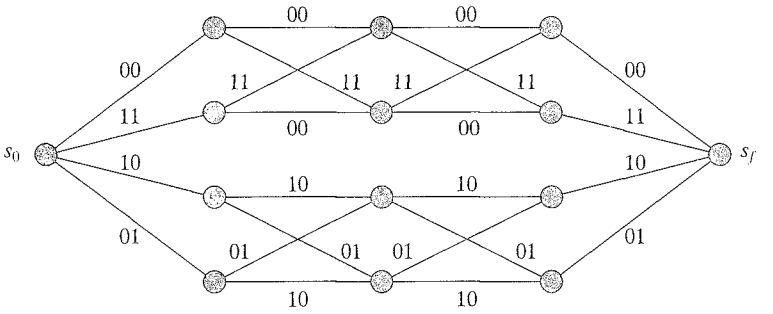

FIGURE 9.17: A 4-section trellis for the (8, 4) RM code.

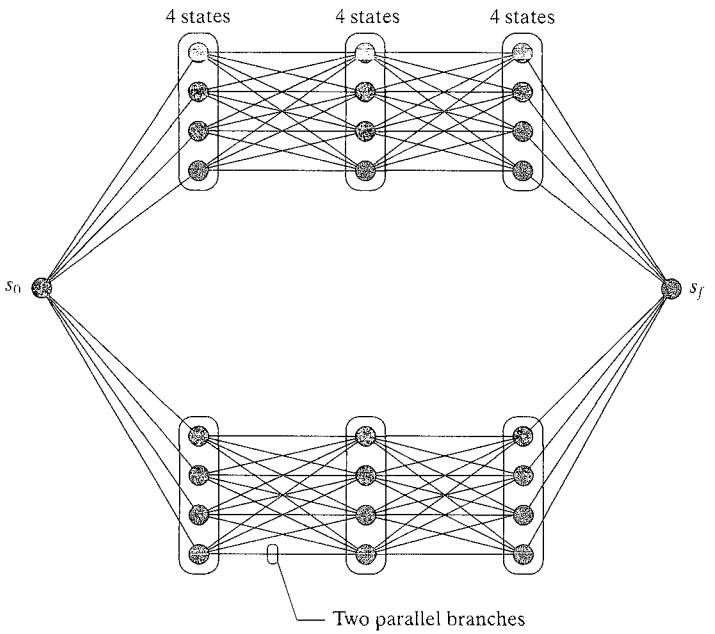

FIGURE 9.18: A 4-section minimal trellis diagram $T(\{0, 4, 8, 12, 16\})$ for the (16, 11) RM code.

4-section trellis as shown in Figure 9.18. Each section is 4 bits long. The state space dimension profile for this 4-section trellis is $(0, 3, 3, 3, 0)$, and the maximum state space dimension is $\rho_{4,\max}(C) = 3$. The trellis consists of two parallel and structurally identical subtrellises without cross-connections between them.

A ν-section trellis obtained from a minimal n-section trellis by deleting states at places other than the section boundary locations is minimal.

A minimal code trellis has the least overall state and branch complexity but is, in general, densely connected. For long codes with large dimensions, it is very difficult to implement any trellis-based MLD algorithm based on a full code trellis on (an) integrated circuit (IC) chip(s) for hardware implementation. To overcome this implementation difficulty, one possible approach is to decompose the minimal trellis diagram of a code into parallel and structurally identical subtrellises of smaller dimensions without cross connections between them so that each subtrellis with reasonable complexity can be put on a single IC chip of reasonable size. This approach is called *parallel decomposition*.

Parallel decomposition should be done in such a way that the maximum state space dimension of the minimal trellis of a code is not exceeded. Parallel decomposition has other advantages. Because all the subtrellises are structurally identical, we can devise identical decoders of much smaller complexity to process these subtrellises in parallel. This not only simplifies the decoding complexity but also speeds up the decoding process. Parallel decomposition also resolves wire

routing problems in IC implementation and reduces internal communications. Wire routing is a major problem in IC implementation of a trellis-based decoder.

Consider the minimal n-section trellis T for a binary (n, k) linear block code C with maximum state space dimension $\rho_{\max}(C)$. We are interested in decomposing T as a disjoint union of a certain desired number of parallel and structurally identical subtrellises under the constraint that its state space dimension at every time-i for $0 \leq i \leq n$ is less than or equal to $\rho_{\max}(C)$; that is, we want to decompose T without exceeding its maximum state complexity.

Suppose we choose a subcode C_1 of C by removing a row $\mathbb{g}$ from the TOGM $\mathbb{G}_{TOGM}$ of C. The generator matrix $\mathbb{G}_1$ for this subcode is $\mathbb{G}_1 = \mathbb{G}_{TOGM} \setminus \{\mathbb{g}\}$. Let $\dim(C)$ and $\dim(C_1)$ denote the dimensions of C and C_1, respectively. Then, $\dim(C_1) = \dim(C) - 1 = k - 1$. The partition C/C_1 consists of two cosets, C_1 and its coset $\mathbb{g} \oplus C_1$. The two coset representatives are generated by $\{\mathbb{g}\}$, which are $\mathbb{0}$ and $\mathbb{g}$. Let T_1 be the minimal trellis for C_1. Recall that the state space dimension $\rho_i(C)$ is equal to the number of rows of $\mathbb{G}$ whose active time spans contain the time index i. Then, we readily see that

$$\rho_i(C_1) = \rho_i(C) - 1 \tag{9.50}$$

for $i \in \tau_a(\mathbb{g})$, and

$$\rho_i(C_1) = \rho_i(C) \tag{9.51}$$

for $i \notin \tau_a(\mathbb{g})$. The equalities of (9.50) and (9.51) give the state space dimension profile of the subcode C_1. We can obtain the minimal trellis T_1' for the coset $\mathbb{g} \oplus C_1$ simply by adding $\mathbb{g}$ to every path in T_1. Therefore, T_1 and T_1' are structurally identical. The union of T_1 and T_1', denoted by $T_1 \cup T_1'$, gives a trellis representation for C in which T_1 and T_1' form two parallel subtrellises without cross connections between them. We may regard that the minimal trellis T for C is decomposed into two parallel and structurally identical subtrellises T_1 and T_1'. Note that $T_1 \cup T_1'$ may not be a minimal trellis for C.

We define the following index set:

$$I_{\max}(C) \triangleq \{i : \rho_i(C) = \rho_{\max}(C), \text{ for } 0 \leq i \leq n\}, \tag{9.52}$$

which is simply the set of time instants (or boundary locations) at which the state space dimensions of the minimal trellis T for C are equal to its maximum, $\rho_{\max}(C)$. Then, Theorem 9.1 follows from (9.50), (9.51), and (9.52) [7, 37, 38].

THEOREM 9.1 If there exists a row $\mathbb{g}$ in the TOGM $\mathbb{G}_{TOGM}$ for an (n, k) linear code C such that $\tau_a(\mathbb{g}) \supseteq I_{\max}(C)$, then the subcode C_1 of C generated by $\mathbb{G}_{TOGM} \setminus \{\mathbb{g}\}$ has a minimal trellis T_1 with maximum state space dimension $\rho_{\max}(C_1) = \rho_{\max}(C) - 1$, and

$$I_{\max}(C_1) = I_{\max}(C) \cup \{i : \rho_i(C) = \rho_{\max}(C) - 1, i \notin \tau_a(\mathbb{g})\}. \tag{9.53}$$

Because $\mathbb{G}$ is in TOF, $\mathbb{G}_1 = \mathbb{G} \setminus \{\mathbb{g}\}$ is also in TOF. If the condition of Theorem 9.1 holds, then it follows from the foregoing that it is possible to construct a trellis for C that consists of two parallel and structurally identical subtrellises, one for C_1 and the other for its coset $C_1 \oplus \mathbb{g}$. Each subtrellis is a minimal trellis and

has maximum state space dimension equal to $\rho_{\max}(C) - 1$. Therefore, the maximum state space dimension of the resultant trellis is still $\rho_{\max}(C)$.

EXAMPLE 9.19

Consider the (8, 4) RM code with TOGM

$$\mathbb{G}_{TOGM} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Its minimal 8-section trellis T is shown in Figure 9.3. Its state space dimension profile is (0, 1, 2, 3, 2, 3, 2, 1, 0) and $\rho_{\max}(C) = 3$. The index set $I_{\max}(C)$ is $I_{\max}(C) = \{3, 5\}$. By examining $\mathbb{G}_{TOGM}$, we find only the second row $\mathbf{g}_1$, whose active time span, $\tau_a(\mathbf{g}_1) = [2, 6]$, contains $I_{\max}(C) = \{3, 5\}$. Suppose we remove $\mathbf{g}_1$ from $\mathbb{G}_{TOGM}$. The resulting matrix is

$$\mathbb{G}_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix},$$

which generates an (8, 3) subcode C_1 of the (8, 4) RM code C. From $\mathbb{G}_1$ we can construct a minimal 8-section trellis T_1 for C_1, as shown in Figure 9.19 (the upper subtrellis). The state space dimension profile of T_1 is (0, 1, 1, 2, 1, 2, 1, 1, 0) and $\rho_{\max}(C) = 2$. Adding $\mathbf{g}_1 = (01011010)$ to every path in T_1, we obtain the minimal 8-section trellis T_1' for the coset $\mathbf{g}_1 \oplus C_1$, as shown in Figure 9.19 (the lower subtrellis). The trellises T_1 and T_1' form a parallel decomposition of the minimal 8-section trellis T for the (8, 4) RM code. We see that the state space dimension profile of $T_1 \cup T_1'$ is (0, 2, 2, 3, 2, 3, 2, 2, 0). Clearly, $T_1 \cup T_1'$ is not a minimal 8-section trellis for the (8, 4) RM code: however, its maximum state space dimension is still $\rho_{\max}(C) = 3$. If we sectionalize $T_1 \cup T_1'$ at locations $\Lambda = \{0, 2, 4, 6, 8\}$, we obtain the minimal 4-section trellis for the code, as shown in Figure 9.17.

We can apply Theorem 9.1 to the subcode C_1 of C if there exists a row $\mathbf{g}' \in \mathbb{G}_1$ such that $\tau_a(\mathbf{g}') \supseteq I_{\max}(C_1)$. Then, $\mathbb{G}_2 = \mathbb{G}_1 \setminus \{\mathbf{g}'\} = \mathbb{G}_{TOGM} \setminus \{\mathbf{g}, \mathbf{g}'\}$ generates a subcode C_2 of C_1 with $\dim(C_2) = \dim(C_1) - 1 = k - 2$. The maximum state space dimension of the minimal trellis for C_2 is then $\rho_{\max}(C_2) = \rho_{\max}(C_1) - 1 = \rho_{\max}(C) - 2$. As a result, the partition C/C_2 decomposes the minimal trellis T for C into four parallel structurally identical subtrellises, one for each coset in C/C_2. The maximum state space dimension of the decomposed trellis is still $\rho_{\max}(C)$.

Theorem 9.1 can be applied repeatedly until either the desired level of decomposition is achieved or no row in the generator matrix can be found to satisfy the condition in Theorem 9.1. A general theorem for parallel decomposition is given next [7, 37, 38].

THEOREM 9.2 Let $\mathbb{G}_{TOGM}$ be the TOGM of an (n, k) linear block code C over $GF(2)$. Define the following subset of rows of $\mathbb{G}_{TOGM}$:

$$R(C) \triangleq \{\mathbf{g} \in \mathbb{G}_{TOGM} : \tau_a(\mathbf{g}) \supseteq I_{\max}(C)\}. \tag{9.54}$$

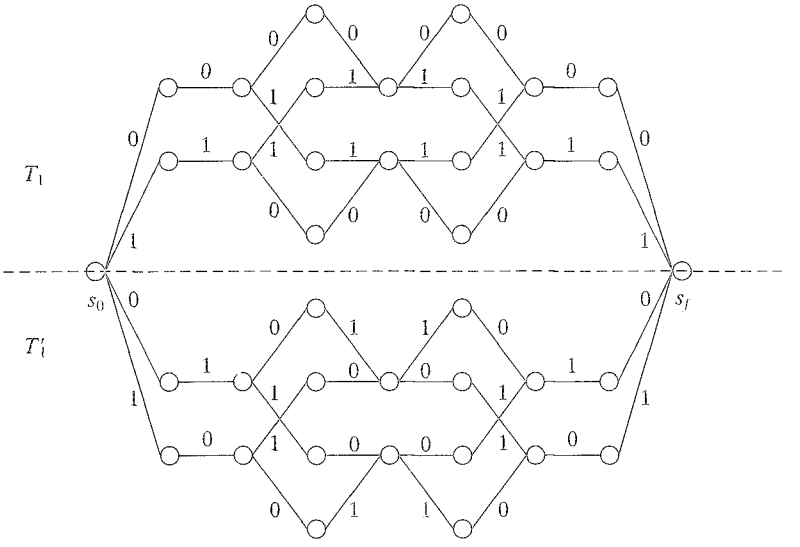

FIGURE 9.19: Parallel decomposition of the minimal 8-section trellis for the (8, 4) RM code.

For any integer r with $1 \leq r \leq |R(C)|$, there exists a subcode of C_r of C such that $\rho_{\max}(C_r) = \rho_{\max}(C) - r$ and $\dim(C_r) = \dim(C) - r$ if and only if there exists a subset $R_r \subseteq R(C)$ consisting of r rows of $R(C)$ such that for every i with $\rho_i(C) > \rho_{\max}(C_r)$, there exist at least $\rho_i(C) - \rho_{\max}(C_r)$ rows in R_r whose active time spans contain i. The subcode C_r is generated by $\mathbb{G}_{TOGM} \setminus R_r$, and the set of coset representatives for C/C_r is generated by R_r.

If the condition of Theorem 9.2 holds, the partition C/C_r results in a parallel decomposition of the minimal trellis T for C that does not exceed the maximum state complexity $\rho_{\max}(C)$ of T. Two direct consequences of Theorem 9.2 are given in Corollaries 9.2.1 and 9.2.2.

COROLLARY 9.2.1 In decomposing a minimal trellis diagram for a linear block code C, the maximum number of parallel isomorphic subtrellises one can obtain such that the total state space dimension at any time does not exceed $\rho_{\max}(C)$ is upper bounded by $2^{|R(C)|}$.

COROLLARY 9.2.2 The logarithm base-2 of the number of parallel isomorphic subtrellises in a minimal v-section trellis with section boundary locations in $\Lambda = \{t_0, t_1, \ldots, t_v\}$ for a binary (n, k) linear block code C is given by the number of rows in the TOGM $\mathbb{G}_{TOGM}$ whose active time spans contain the section boundary locations, $t_1, t_2, \ldots, t_{v-1}$.

EXAMPLE 9.20

Consider the (8, 4) RM code given in Example 9.1, whose minimal 4-section trellis diagram with section boundary locations in $\Lambda = \{0, 2, 4, 6, 8\}$ is shown in Figure 9.17.

By examining the TOGM $\mathbf{G}_{TOGM}$ of the code, we find only one row, the second row, $\mathbf{g}_1 = (0\,1\,0\,1\,1\,1\,0\,1\,0)$, whose active time span, $\tau_a(\mathbf{g}_1) = [2, 6]$, contains $\{2, 4, 6\}$. Therefore, the minimal 4-section trellis consists of two parallel isomorphic subtrellises.

EXAMPLE 9.21

Consider the (16, 11) RM code given in Example 9.10. Suppose we sectionalize the bit-level trellis at locations in $\Lambda = \{0, 4, 8, 12, 16\}$. Examining the TOGM of the code, we find only row $\mathbf{g}_3 = (0\,0\,0\,1\,1\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0)$ whose active time span, $\tau_a(\mathbf{g}_3) = [4, 12]$, contains $\{4, 8, 12\}$. Therefore, the 4-section trellis $T(\{0, 4, 8, 12, 16\})$ consists of two parallel isomorphic subtrellises.

Basically, Theorem 9.2 provides a method for decomposing a complex minimal trellis into subtrellises without exceeding the maximum state complexity. This decomposition of a trellis into parallel and structurally identical subtrellises of smaller state complexity without cross connections between them has significant advantages for IC implementation of a trellis-based decoding algorithm, such as the Viterbi algorithm to be discussed in Chapters 12 and 15. Identical decoders of much simpler complexity can be devised to process the subtrellises independently in parallel without internal communications (or information transfer) between them. Internal information transfer limits the decoding speed. Furthermore, the number of computations to be carried out per subtrellis is much smaller than that of a fully connected trellis. As a result, the parallel structure not only simplifies the decoding complexity but also speeds up the decoding process.

9.8 LOW-WEIGHT SUBTRELLISES

Consider a binary (n, k) linear block code C with weight profile $W = \{0, w_1, w_2, \ldots, w_m\}$, where w_1 is the minimum weight and $w_1 < w_2 < \ldots < w_m \leq n$. Let $T(\Lambda)$ be a ν-section trellis for C with section boundary locations in $\Lambda = \{t_0 = 0, t_1, t_2, \ldots, t_{\nu-1}, t_\nu = n\}$. A subtrellis of $T(\Lambda)$ that consists of the all-zero codeword $\mathbb{0}$ and all the codewords of C with weights up to w_k (including w_k) is called the *w_k-weight subtrellis* of $T(\Lambda)$, denoted by $T_{w_k}(\mathbb{0})$. This subtrellis is said to be centered at $\mathbb{0}$. For small k, say $k = 1$ or 2, this subtrellis is called a *low-weight subtrellis*. The w_k-weight subtrellis $T_{w_k}(\mathbb{0})$ can be obtained by purging the full trellis $T(\Lambda)$ by using the algorithm described in [7, 37, 39].

For two adjacent states s and s', with $s \in \Sigma_{t_j}(C)$ and $s' \in \Sigma_{t_{j+1}}(C)$, let $L(s, s')$ denote the set of parallel branches connecting the states s and s'. Let $\mathbb{b} \in L(s, s')$ and $w(\mathbb{b})$ denote the Hamming weight of $\mathbb{b}$. Let $\alpha(L(s, s'))$ denote the minimum weight of the parallel branches in $L(s, s')$. For any state $s \in \Sigma_{t_j}(C)$ with $0 \leq j \leq \nu$, let $W_I(s)$ denote the minimum of the weights of all paths from the initial state s_0 to the state s. $W_I(s)$ is called the *minimum path weight* to the state s. For every state s in $T(\Lambda)$, $W_I(s)$ can be computed recursively. Suppose that $W_I(s)$ is known for every state $s \in \Sigma_{t_i}(C)$ for $0 \leq i \leq j$. Let $s' \in \Sigma_{t_{j+1}}(C)$ and $F(s')$ denote the set of states in $\Sigma_{t_j}(C)$ that are adjacent to s'. Then,

$$W_I(s') = \min_{s \in F(s')} \{W_I(s) + \alpha(L(s, s'))\}. \tag{9.55}$$

The recursion begins with $W_I(s_0) = 0$. Once $W_I(s)$ is determined for every state $s \in \Sigma_{t_j}(C)$, the states in $\Sigma_{t_{j+1}}(C)$ can be processed. The weight computation of (9.55) continues until $W(s_f)(= 0)$ is determined.

For any state $s \in \Sigma_{t_j}(C)$ with $0 \leq j \leq \nu$, let $W_F(s)$ denote the minimum of the weights of all paths from state s to the final state s_f. $W_F(s)$ is called the minimum path weight from the state s. For every state s in $T(\Lambda)$, $W_F(s)$ can also be computed recursively from the final state s_f. Suppose $W_F(s)$ is known for every state $s \in \Sigma_{t_i}(C)$ for $j < i \leq \nu$. Let s' be a state in $\Sigma_{t_j}(C)$, and let $G(s')$ denote the set of states in $\Sigma_{t_{j+1}}(C)$ that are adjacent to s'. Then,

$$W_F(s') = \min_{s \in G(s')} \{W_F(s) + \alpha(L(s', s))\}. \tag{9.56}$$

It follows from the definitions of $W_I(s)$ and $W_F(s)$ that there is at least one path in $T(\Lambda)$ passing through state s with weight $W_I(s) + W_F(s)$, and there is no path in $T(\Lambda)$ passing through state s with weight less than $W_I(s) + W_F(s)$. Let the all-zero path $\mathbb{0}$ in $T(\Lambda)$ be represented by the state sequence

$$s_0 = s_0^{(\mathbb{0})}, s_{t_1}^{(\mathbb{0})}, s_{t_2}^{(\mathbb{0})}, \ldots, s_{t_\nu}^{(\mathbb{0})} = s_f.$$

Then, $W_I(s_{t_j}^{(\mathbb{0})}) + W_F(s_{t_j}^{(\mathbb{0})}) = 0$ for $0 \leq j \leq \nu$ and for any state s not on the all-zero path $\mathbb{0}$, $W_I(s) + W_F(s) > 0$.

The w_k-weight subtrellis $T_{w_k}(\mathbb{0})$ can be obtained by purging the full trellis $T(\Lambda)$ as follows:

1. If for every state $s \in T(\Lambda)$, $W_I(s) + W_F(s) > w_k$, delete state s and all branches entering and leaving s.
2. Let s and s' be any two adjacent states with $s \in \Sigma_{t_j}(C)$ and $s' \in \Sigma_{t_{j+1}}(C)$ for $0 \leq j < \nu$. Let $\mathbb{b} \in L(s, s')$. If $W_I(s) + W_F(s') + w(\mathbb{b}) > w_k$, delete branch $\mathbb{b}$.
3. If as a result of applications of the preceding two purging steps, any state $s(\neq s_0)$ has no incoming branches, then delete s and all its outgoing branches from $T(\Lambda)$. If any state $s(\neq s_f)$ has no outgoing branches, then delete s and all its incoming branches from $T(\Lambda)$.

The foregoing purging process continues until none of the three purging steps can be applied. The resultant subtrellis contains the following codewords of C: (1) all codewords of weights up to and including w_k; and (2) possibly some codewords of weights greater than w_k that correspond to nonzero paths in $T(\Lambda)$ that diverge from and remerge with the all-zero path $\mathbb{0}$ more than once. For each of these paths, the weight of a partial path between the adjacent diverging and remerging states, called *side-loop*, is w_k or less but not zero. The number of these paths is, in general, small and they can be removed by using the state-splitting technique [7, 40]. We call the described purged trellis the w_k-weight subtrellis of $T(\Lambda)$.

For $k = 1$, $T_{w_1}(\mathbb{0})$ contains (1) the all-zero codeword $\mathbb{0}$; (2) all the codewords of minimum weight w_1; and (3) those codewords of weights greater than w_1 that correspond to nonzero paths that diverge from and remerge with the all-zero path more than once. For each of these nonzero paths, the weight of each side-loop is w_1. These paths with weights greater than w_1 can be removed by splitting the states

$$s_{t_1}^{(\mathbb{0})}, s_{t_2}^{(\mathbb{0})}, \ldots, s_{t_{\nu-1}}^{(\mathbb{0})}$$

on the all-zero path $\mathbb{0}$ as follows:

1. Create $\nu - 1$ new states, denoted by $\tilde{s}_{t_1}^{(\mathbb{0})}, \tilde{s}_{t_2}^{(\mathbb{0})}, \ldots, \tilde{s}_{t_{\nu-1}}^{(\mathbb{0})}$, one at each boundary location except $t_0 = 0$ and $t_\nu = n$;
2. For $1 \leq j < \nu - 1$, connect the state $\tilde{s}_{t_j}^{(\mathbb{0})}$ to state $\tilde{s}_{t_{j+1}}^{(\mathbb{0})}$ by a single branch with the all-zero $(t_{j+1} - t_j)$-tuple label. Also connect $\tilde{s}_{t_{\nu-1}}^{(\mathbb{0})}$ to the final state s_f by a single all-zero label; and
3. Delete every branch $\mathbb{b}$ with nonzero weight (i.e., $w(\mathbb{b}) \neq 0$) that merges into one of the states, $s_{t_j}^{(\mathbb{0})}$, on the all-zero path from any state s, and create a new branch with the same label from s to state $\tilde{s}_{t_j}^{(\mathbb{0})}$.

The foregoing state-splitting process ensures that there is no path in the new trellis that after merging with the all-zero path diverges from it again before terminating at the final state, s_f. Consequently, the new trellis, denoted by $T_{\min}(\mathbb{0})$ contains only the all-zero path $\mathbb{0}$ and the minimum weight codewords of C. $T_{\min}(\mathbb{0})$ is called the *minimum-weight subtrellis* of C and is said to be centered at $\mathbb{0}$.

The subtrellis that contains a nonzero codeword $\mathbb{v}$ of C and all the codewords that are at minimum distance from $\mathbb{v}$, denoted by $T_{\min}(\mathbb{v})$, can be obtained by adding $\mathbb{v}$ to all the codewords in $T_{\min}(\mathbb{0})$. This subtrellis $T_{\min}(\mathbb{v})$ is called the minimum-weight (or more accurately minimum distance) subtrellis of C centered at $\mathbb{v}$.

The minimum-weight trellis $T_{\min}(\mathbb{0})$ is sparsely connected and generally has much smaller state and branch complexities than does the full trellis for the code. For example, the state space complexity profile for the 4-section minimum-weight trellis for the (64, 42) RM code is (1, 157, 157, 157, 1) as compared with (1, 1024, 1024, 1024, 1) for the 4-section full trellis of the code. For the same code, the 8-section minimum-weight subtrellis has state complexity profile

$$(1, 45, 157, 717, 157, 717, 157, 45, 1)$$

as compared with

$$(1, 128, 1024, 8192, 1024, 8192, 1024, 128, 1)$$

for the 8-section full trellis of the code. For this code, the 8-section minimum-weight subtrellis has a total of 4524 branches as compared with a total of 278, 784 branches for the 8-section full trellis.

9.9 CARTESIAN PRODUCT

A trellis for a linear block code constructed from component codes by using a construction technique such as interleaving, product, or direct-sum can be constructed by taking *Cartesian product* of the trellises of the component codes.

Consider the interleaved code $C^\lambda = C_1 * C_2 * \cdots * C_\lambda$, which is constructed by interleaving λ linear block codes, $C_1, C_2, \cdots, C_\lambda$, of length n (see Section 4.8). An n-section trellis T^λ can be constructed by taking the Cartesian product of the n-section trellises of the component codes. For $1 \leq j \leq \lambda$, let T_j be an n-section trellis for C_j. For $0 \leq i \leq n$, let $\Sigma_i(C_j)$ denote the state space of T_j at time-i. The Cartesian product of $T_1, T_2, \cdots, T_\lambda$, denoted by $T^\lambda \triangleq T_1 \times T_2 \times \cdots \times T_\lambda$, is constructed as follows:

1. For $0 \le i \le n$, form the Cartesian product of $\Sigma_i(C_1), \Sigma_i(C_2), \cdots, \Sigma_i(C_\lambda)$.

$$\begin{aligned}\Sigma_i(C^\lambda) &\triangleq \Sigma_i(C_1) \times \Sigma_i(C_2) \times \cdots \times \Sigma_i(C_\lambda) \\ &= \{(s_i^{(1)}, s_i^{(2)}, \cdots, s_i^{(\lambda)}) : s_i^{(j)} \in \Sigma_i(C_j) \text{ for } 1 \le j \le \lambda\}.\end{aligned} \tag{9.57}$$

   Then, $\Sigma_i(C^\lambda)$ forms the state space of T^λ at time-i, i.e., the λ-tuples in $\Sigma_i(C^\lambda)$ form the nodes of T^λ at level-i.

2. A state $(s_i^{(1)}, s_i^{(2)}, \cdots, s_i^{(\lambda)})$ in $\Sigma_i(C^\lambda)$ is adjacent to a state $(s_{i+1}^{(1)}, s_{i+1}^{(2)}, \cdots, s_{i+1}^{(\lambda)})$ in $\Sigma_{i+1}(C^\lambda)$ if and only if $s_i^{(j)}$ is adjacent to $s_{i+1}^{(j)}$ for $1 \le j \le \lambda$. Let $l_j \triangleq l(s_i^{(j)}, s_{i+1}^{(j)})$ denote the label of the branch that connects the state $s_i^{(j)}$ to the state $s_{i+1}^{(j)}$ for $1 \le j \le \lambda$. We connect the state $(s_i^{(1)}, s_i^{(2)}, \cdots, s_i^{(\lambda)}) \in \Sigma_i(C^\lambda)$ to the state $(s_{i+1}^{(1)}, s_{i+1}^{(2)}, \cdots, s_{i+1}^{(\lambda)}) \in \Sigma_{i+1}(C^\lambda)$ by a branch that is labeled by the following λ-tuple:

$$(l_1, l_2, \cdots, l_\lambda).$$

   This label is simply a column of the array of (4.80).

The constructed Cartesian product $T^\lambda = T_1 \times T_2 \times \cdots \times T_\lambda$ is an n-section trellis for the interleaved code $C^\lambda = C_1 * C_2 * \cdots * C_\lambda$ in which each section is of length λ.

EXAMPLE 9.22

Let C be the $(3, 2)$ even parity-check code whose generator matrix in trellis-oriented form is

$$\mathbb{G}_{TOGM} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

The 3-section bit-level trellis T for this code can easily be constructed from $\mathbb{G}_{TOGM}$ and is shown in Figure 9.20(a). Suppose the code is interleaved to a depth of $\lambda = 2$. Then, the interleaved code $C^2 = C * C$ is a $(6, 4)$ linear code. The Cartesian product $T \times T$ results in a 3-section trellis T^2 for C^2, as shown in Figure 9.20(b).

Let C_1 be an (n_1, k_1) linear code, and let C_2 be an (n_2, k_2) linear code. The product $C_1 \times C_2$ is then an $(n_1 n_2, k_1 k_2)$ linear block code (see Section 4.7). To construct a trellis for the product code $C_1 \times C_2$, we regard the top k_2 rows of the product array shown in Figure 4.3 as an interleaved array with codewords from the same code C_1. We then construct an n_1-section trellis for the interleaved code,

$$C_1^{k_2} = \underbrace{C_1 * C_1 * \cdots * C_1}_{k_2},$$

using the Cartesian product. Each k_2-tuple branch label in the trellis is encoded into a codeword in C_2. The result is an n_1-section trellis for the product $C_1 \times C_2$, each section is n_2-bit in length.

EXAMPLE 9.23

Let C_1 and C_2 both be the $(3, 2)$ even parity-check code. Then, the product $C_1 \times C_2$ is a $(9, 4)$ linear code with a minimum distance of 4. Using the Cartesian product

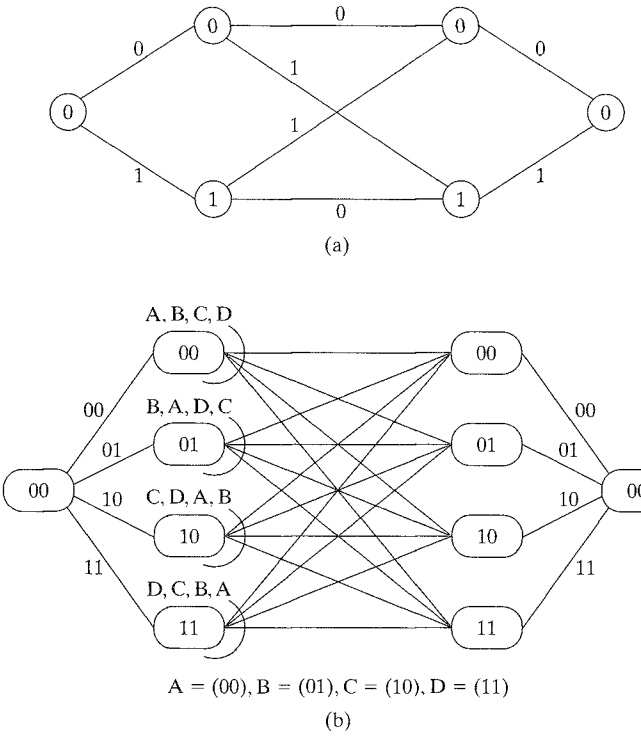

FIGURE 9.20: (a) The minimal 3-section bit-level trellis for the (3, 2) even parity-check code, (b) A 3-section trellis for the interleaved (3, 2) code of depth 2.

construction method given above, we first construct the 3-section trellis for the interleaved code $C_1^2 = C_1 * C_1$, which is shown in Figure 9.20(b). Then, we encode each branch label in this trellis based on the $C_2 = (3, 2)$ code. The result is a 3-section trellis for the product code $C_1 \times C_2$, as shown in Figure 9.21.

Let C_1 and C_2 be an (n, k_1, d_1) and an (n, k_2, d_2) binary linear block code with generator matrix, $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Suppose C_1 and C_2 have only the all-zero codeword $\mathbb{0}$ in common; that is, $C_1 \cap C_2 = \{\mathbb{0}\}$. Their direct-sum, denoted by $C_1 \oplus C_2$, is defined as follows:

$$C \triangleq C_1 \oplus C_2 \triangleq \{\mathbb{u} + \mathbb{v} : \mathbb{u} \in C_1, \mathbb{v} \in C_2\}. \tag{9.58}$$

Then, $C = C_1 \oplus C_2$ is an $(n, k_1 + k_2)$ code with minimum distance $d_{\min} \leq \min\{d_1, d_2\}$ and generator matrix

$$\mathbb{G} = \begin{bmatrix} \mathbb{G}_1 \\ \mathbb{G}_2 \end{bmatrix}.$$

Let T_1 and T_2 be the n-section trellis for C_1 and C_2, respectively. Then, an n-section trellis T for the direct-sum code $C = C_1 \oplus C_2$ can be constructed by taking the Cartesian product of T_1 and T_2. The formation of state spaces for T and the condition for state adjacency between states in T are the same as for forming the trellis for an

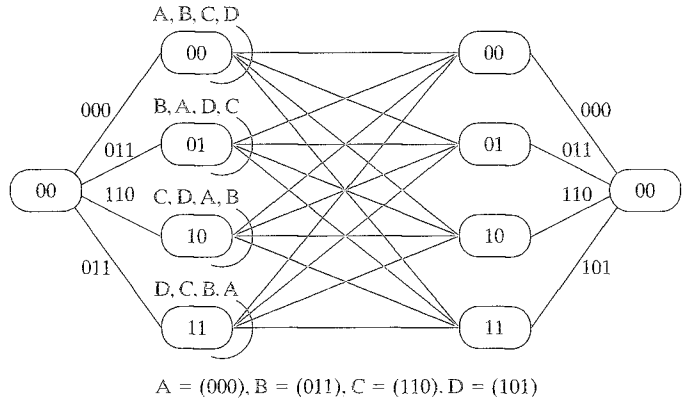

FIGURE 9.21: A 3-section trellis for the product $(3, 2) \times (3, 2)$.

interleaved code by taking the Cartesian product of the trellises for the component codes. The difference is branch labeling. For two adjacent states $(s_i^{(1)}, s_i^{(2)})$ and $(s_{i+1}^{(1)}, s_{i+1}^{(2)})$ in T at time-i and time-$(i+1)$, let $l_j \triangleq l(s_i^{(j)}, s_{i+1}^{(j)})$ be the label of the branch that connects the state $s_i^{(j)}$ and the state $s_{i+1}^{(j)}$ for $1 \leq j \leq 2$. We connect the state $(s_i^{(1)}, s_i^{(2)})$ and the state $(s_{i+1}^{(1)}, s_{i+1}^{(2)})$ with a branch that is labeled with $l_1 + l_2 = l(s_i^{(1)}, s_{i+1}^{(1)}) + l(s_i^{(2)}, s_{i+1}^{(2)})$. The described product of two trellises is also known as the *Shannon product*.

EXAMPLE 9.24

Let C_1 and C_2 be two linear block codes of length 8 generated by

$$\mathbb{G}_1 = \begin{bmatrix} 1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,1\,0\,1\,1\,0\,1\,0 \end{bmatrix}$$

and

$$\mathbb{G}_2 = \begin{bmatrix} 0\,0\,1\,1\,1\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,1\,1 \end{bmatrix},$$

respectively. It is easy to check that $C_1 \cap C_2 = \{\mathbb{0}\}$. The direct-sum $C_1 \oplus C_2$ is generated by

$$\mathbb{G} = \begin{bmatrix} \mathbb{G}_1 \\ \mathbb{G}_2 \end{bmatrix} = \begin{bmatrix} 1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,1\,0\,1\,1\,0\,1\,0 \\ 0\,0\,1\,1\,1\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,1\,1 \end{bmatrix},$$

which is simply the TOGM for the $(8, 4, 4)$ RM code given in Example 9.1. Both $\mathbb{G}_1$ and $\mathbb{G}_2$ are in TOF. Based on $\mathbb{G}_1$ and $\mathbb{G}_2$, we construct the 8-section trellises T_1 and T_2 for C_1 and C_2 as shown in Figures 9.22 and 9.23, respectively. Taking the Shannon product of T_1 and T_2, we obtain an 8-section trellis $T_1 \times T_2$ for the direct-sum $C_1 \oplus C_2$ as shown in Figure 9.24, which is simply the 8-section minimal trellis for the $(8, 4, 4)$ RM code as shown in Figure 9.6. We label the states in T_1, T_2 and $T_1 \times T_2$ by the method of state-defining sets of Section 9.3.

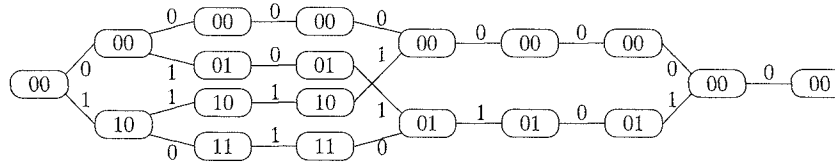

FIGURE 9.22: The 8-section minimum trellis for the code generated by $\mathbb{G}_1 = \begin{bmatrix} 1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,1\,0\,1\,1\,0\,1\,0 \end{bmatrix}$.

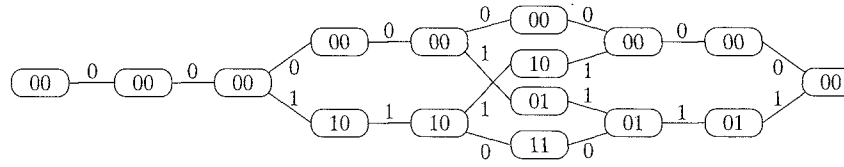

FIGURE 9.23: The 8-section minimum trellis for the code generated by $\mathbb{G}_2 = \begin{bmatrix} 0\,0\,1\,1\,1\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,1\,1 \end{bmatrix}$.

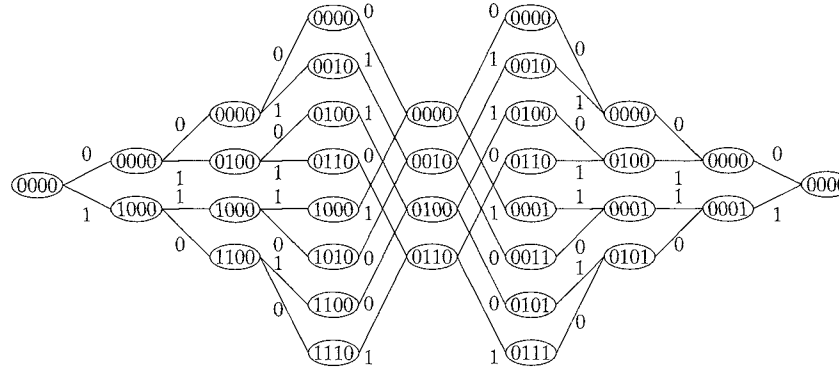

FIGURE 9.24: The Shannon product of the trellises of Figures 9.22 and 9.23.

The Shannon product can be generalized to construct a trellis for a code that is a direct-sum of m linear block codes. For a positive integer $m \geq 2$, and $1 \leq j \leq m$, let C_j be an (N, K_j, d_j) linear code. Suppose $C_1, C_2, \cdots, C_m$ satisfy the following condition: for $1 \leq j, j' \leq m$, and $j \neq j'$,

$$C_j \cap C_{j'} = \{\mathbb{0}\}. \tag{9.59}$$

This condition simply implies that for $\mathbb{v}_j \in C_j$ with $1 \leq j \leq m$,

$$\mathbb{v}_1 + \mathbb{v}_2 + \cdots + \mathbb{v}_m = \mathbb{0}$$

if and only if $\mathbb{v}_1 = \mathbb{v}_2 = \cdots = \mathbb{v}_m = \mathbb{0}$. The direct-sum of $C_1, C_2, \cdots, C_m$ is defined as

$$\begin{aligned} C &\triangleq C_1 \oplus C_2 \oplus \cdots \oplus C_m \\ &= \{\mathbb{v}_1 + \mathbb{v}_2 + \cdots + \mathbb{v}_m : \mathbb{v}_j \in C_j, 1 \le j \le m\}. \end{aligned} \tag{9.60}$$

Then, $C = C_1 \oplus C_2 \oplus \cdots \oplus C_m$ is an (N, K, d) linear block code with

$$\begin{aligned} K &= K_1 + K_2 + \cdots + K_m \\ d &\le \min_{1 \le j \le m} \{d_j\}. \end{aligned}$$

Let $\mathbb{G}_j$ be the generator matrix of C_j for $1 \le j \le m$. Then, $C = C_1 \oplus C_2 \oplus \cdots \oplus C_m$ is generated by the following matrix:

$$\mathbb{G} = \begin{bmatrix} \mathbb{G}_1 \\ \mathbb{G}_2 \\ \vdots \\ \mathbb{G}_m \end{bmatrix}. \tag{9.61}$$

The construction of an n-section trellis T for the direct-sum $C = C_1 \oplus C_2 \oplus \cdots \oplus C_m$ is the same as that for the direct-sum of two codes. Let $(s_i^{(1)}, s_i^{(2)}, \cdots, s_i^{(m)})$ and $(s_{i+1}^{(1)}, s_{i+1}^{(2)}, \cdots, s_{i+1}^{(m)})$ be two adjacent states in T. Then, the branch connecting $(s_i^{(1)}, s_i^{(2)}, \cdots, s_i^{(m)})$ to $(s_{i+1}^{(1)}, s_{i+1}^{(2)}, \cdots, s_{i+1}^{(m)})$ is labeled with

$$l(s_i^{(1)}, s_{i+1}^{(1)}) + l(s_i^{(2)}, s_{i+1}^{(2)}) + \cdots + l(s_i^{(m)}, s_{i+1}^{(m)}), \tag{9.62}$$

where for $1 \le j \le m$, $l(s_i^{(j)}, s_{i+1}^{(j)})$ is the label of the branch that connects the state $s_i^{(j)}$ and the state $s_{i+1}^{(j)}$ in the trellis T_j for the jth code C_j.

EXAMPLE 9.25

Again, we consider the (8, 4, 4) RM code generated by the following TOGM:

$$\mathbb{G} = \begin{bmatrix} 1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,1\,0\,1\,1\,0\,1\,0 \\ 0\,0\,1\,1\,1\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,1\,1 \end{bmatrix}.$$

For $1 \le j \le 4$, let C_j be the (8, 1, 4) code generated by the jth row of $\mathbb{G}$. Then, the direct-sum, $C_1 \oplus C_2 \oplus C_3 \oplus C_4$, gives the (8, 4, 4) RM code. The 8-section minimal trellises for the four component codes are shown in Figure 9.25. The Shannon products $T_1 \times T_2$ and $T_3 \times T_4$ generate the trellises for $C_1 \oplus C_2$ and $C_3 \oplus C_4$, respectively, as shown in Figures 9.22 and 9.23. The Shannon product $(T_1 \times T_2) \times (T_3 \times T_4)$ results in the overall trellis for the (8, 4, 4) RM code shown in Figure 9.24.

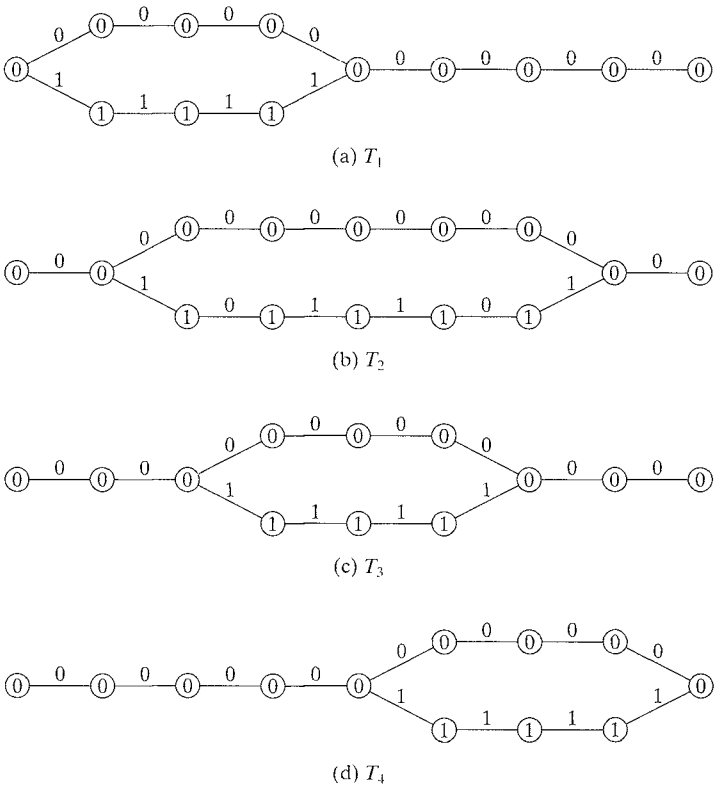

FIGURE 9.25: The 8-section minimal trellises for the four component codes.

Suppose $T_1, T_2, \cdots, T_m$ are minimal N-section trellises for the component codes $C_1, C_2, \cdots, C_m$, respectively. The Shannon product $T_1 \times T_2 \times \cdots \times T_m$ is not necessarily the minimal n-section trellis for the direct-sum $C = C_1 \oplus C_2 \oplus \cdots \oplus C_m$. Let T denote the minimal n-section trellis for C, and $\rho_i(C)$ denote the state space dimension of T at time-i for $0 \leq i \leq n$. Then,

$$\rho_i(C) \leq \sum_{j=1}^{m} \rho_i(C_j). \tag{9.63}$$

If the equality of (9.63) holds for $0 \leq i \leq n$, then the Shannon product $T_1 \times T_2 \times \cdots \times T_m$ is the minimal n-section trellis for the direct-sum $C = C_1 \oplus C_2 \oplus \cdots \oplus C_m$. Theorem 9.3 gives a sufficient condition for the equality of (9.63). A proof of this theorem can be found in [7].

THEOREM 9.3 Consider the direct-sum $C = C_1 \oplus C_2 \oplus \cdots \oplus C_m$. For $1 \leq j \leq m$, let T_j be the minimal N-section trellis for the component code C_j. Then, the Shannon product $T_1 \times T_2 \times \cdots \times T_m$ is the minimal N-section trellis for C if and only if the following condition holds: for $0 \leq i \leq N$, $1 \leq j, j' \leq m$, and $j \neq j'$,

$$p_{0,i}(C_j) \cap p_{0,i}(C_{j'}) = \{\mathbf{0}\}. \tag{9.64}$$

The Shannon product can be applied to sectionalized trellises. In this case, all the trellises must have the same number of sections, and corresponding sections must have the same length.

EXAMPLE 9.26

Suppose we sectionalize each of the two 8-section trellises of Figures 9.22 and 9.23 into 4 sections, each of length 2. The resultant 4-section trellises are shown in

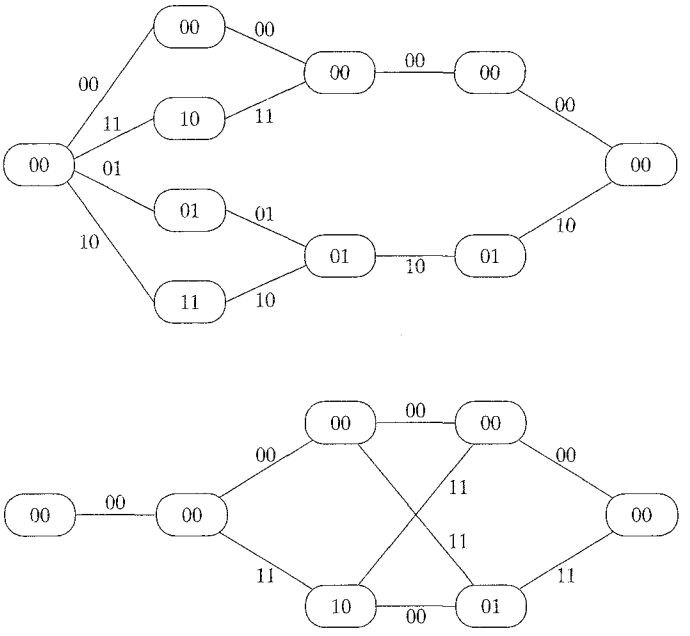

FIGURE 9.26: The 4-section trellises for the trellises of Figures 9.22 and 9.23.

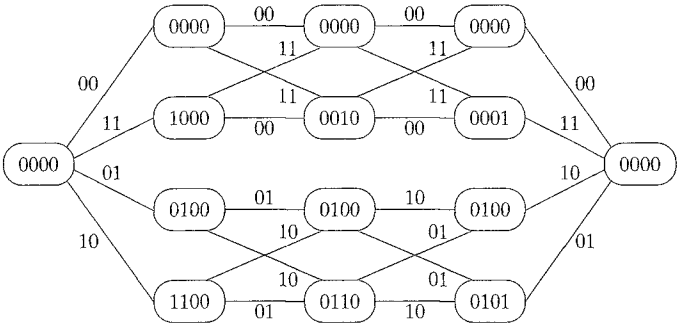

FIGURE 9.27: A 4-section trellis for the (8, 4, 4) RM code.

Figure 9.26, and the Shannon product of these two 4-section trellises gives a 4-section trellis, as shown in Figure 9.27, which is the same as the 4-section trellis for the (8, 4, 4) RM code shown in Figure 9.17.

PROBLEMS

9.1 Consider the (6, 3) linear code generated by the following matrix:

$$\mathbf{G} = \begin{bmatrix} 1\,0\,1\,1\,0\,1 \\ 1\,1\,0\,0\,0\,1 \\ 1\,0\,1\,0\,1\,0 \end{bmatrix}.$$

a. Put this generator in trellis-oriented form.
b. Determine the active time spans of the rows in the trellis-oriented generator matrix.
c. Determine the state space dimension profile of the bit-level 6-section trellis for the code.
d. Determine the state-defining information set at each time instant.
e. Determine the input information bit at each time instant.
f. Determine the output function in each bit interval.

9.2 Construct the trellis-oriented generator matrix for the first-order RM code, RM(1, 5), of length 32.
a. Determine the active time spans of the rows.
b. Determine the state space dimension profile of the bit-level trellis for the code.
c. Determine the state-defining information set at each time instant.
d. Determine the input information bit at each time instant.
e. Determine the output function in each bit interval.

9.3 Construct the bit-level trellis for the (6, 3) code given in Problem 9.1. Label the states based on the state-defining information set using $\rho_{\max}(C)$ bits.

9.4 Find a parity-check matrix for the (6, 3) code given in Problem 9.1. Label the states of its bit-level trellis based on the parity-check matrix.

9.5 Construct the bit-level minimal trellis for the (8, 7) even-parity-check code.

9.6 Construct the bit-level trellis for the first-order RM code, RM(1, 5), of length 32. Label its states based on the state-defining information set using $\rho_{\max}(C)$ bits.

9.7 Determine the past and future subcodes of the (6, 3) linear code given in Problem 9.1 at each time instant. Determine the cosets in the partition

$$C/C_{0,4} \oplus C_{4,6}.$$

9.8 Determine the past and future subcodes of the first-order RM code, RM(1, 4), of length 16 at time instants 4, 8, and 12. Determine the cosets in the partition

$$C/C_{0,8} \oplus C_{8,16}.$$

9.9 For the first-order RM code of length 16, determine the punctured code $p_{4,8}(C)$ and punctured code $C^{tr}_{4,8}$ between time-4 and time-8. Determine the partition

$$p_{4,8}(C)/C^{tr}_{4,8}.$$

9.10 Determine the state space dimension profile of the bit-level trellis for the primitive (15,5) BCH code. Construct its bit-level trellis.

9.11 Consider the first-order RM code of length 16 given in Example 9.13. Construct a 4-section trellis for the code with section boundary locations at 0, 4, 8, 12, and 16.

9.12 Continue Problem 9.5. Construct a 4-section trellis for the first-order RM code of length 32.

9.13 Consider the first-order RM code of length 16 given in Example 9.13. Decompose the bit-level trellis into two parallel subtrellises without exceeding the maximum state space dimension.

9.14 Continue Problem 9.13. After decomposition, construct an 8-section trellis for the code.

9.15 Can the first-order RM code of length 16 be decomposed into 4 parallel subtrellises without exceeding its maximum state space dimension?

9.16 Can the bit-level trellis for the primitive (15, 5) BCH code be decomposed into two parallel subtrellises without exceeding its maximum state space dimension? If yes, decompose the trellis.

9.17 Prove that the bit-level trellis for the first-order RM code of length 16 has mirror-image symmetry.

9.18 Prove that the bit-level trellis for the first-order RM code, RM(r, m), has mirror-image symmetry.

BIBLIOGRAPHY

1. G. D. Forney, Jr., "The Viterbi Algorithm," *Proc. IEEE*, 61: 268–78, 1973.

2. A. J. Viterbi, "Error Bounds for Convolutional Codes and Asymptotically Optimum Decoding Algorithm," *IEEE Trans. Inform. Theory*, IT-13: 260–69, 1967.

3. L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. Inform. Theory*, IT-20: 284–87, 1974.

4. J. K. Wolf, " Efficient Maximum-Likelihood Decoding of Linear Block Codes Using a Trellis," *IEEE Trans. Inform. Theory*, IT-24: 76–80, 1978.

5. J. L. Massey, "Foundation and Methods of Channel Encoding, " in *Proc. Int. Conf. Inform. Theory and Systems*, NTG-Fachberichte, Berlin, 1978.

6. G. D. Forney Jr., "Coset Codes II: Binary Lattices and Related Codes," *IEEE Trans. Inform. Theory*, IT-34: 1152–87, 1988.

7. S. Lin, T. Kasami, T. Fujiwara, and M. P. C. Fossorier, *Trellises and Trellis-Based Decoding Algorithms for Linear Block Codes*, Kluwer Academic, Boston, Mass., 1998.

8. A. Vardy, "Trellis Structure of Codes" in *Handbook of Coding Theory*, edited by V. Pless, W. Huffman, and R. A. Brualdi, Elsevier Science, Amsterdam, 1998.

9. D. J. Muder, "Minimal Trellises for Block Codes," *IEEE Trans. Inform. Theory*, 34: 1049–53, 1988.

10. Y. Berger and Y. Be'ery, "Bounds on the Trellis Size of Linear Block Codes," *IEEE Trans. Inform. Theory*, IT-39: 203–9, 1993.

11. T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On the Optimum Bit Orders with Respect to the State Complexity of Trellis Diagrams for Binary Linear Codes," *IEEE Trans. Inform. Theory*, IT-39: 242–43, January 1993.

12. T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On Complexity of Trellis Structure of Linear Block Codes," *IEEE Trans. Inform. Theory*, IT-39: 1057–64, May 1993.

13. T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On Structural Complexity of the L-section Minimal Trellis Diagrams for Binary Linear Block Codes," *IEICE Trans. Fundamentals*, E76-A (9): 1411–21, September 1993.

14. G. D. Forney, Jr., and M. D. Trott, "The Dynamics of Group Codes: State Spaces, Trellis Diagrams and Canonical Encoders," *IEEE Trans. Inform. Theory*, IT-39: 1491–513, 1993.

15. T. Kasami, T. Fujiwara, Y. Desaki, and S. Lin, "On Branch Labels of Parallel Components of the L-Section Minimal Trellis Diagrams for Binary Linear Block Codes," *IEICE Trans. Fundamentals*, E77-A (6): 1058–68, June 1994.

16. G. D. Forney, Jr., "Dimension/Length Profiles and Trellis Complexity of Linear Block Codes," *IEEE Trans. Inform. Theory*, IT-40: 1741–52, 1994.

17. S. Dolinar, L. Ekroot, A. B. Kiely, R. J. McEliece, and W. Lin, "The Permutation Trellis Complexity of Linear Block Codes," in *Proc. 32nd Allerton Conf. on Comm., Control, and Computing*, 60–74, Monticello, Ill. September 1994.

18. F. R. Kschischang, "The Combinatorics of Block Code Trellises," in *Proc. Biennial Symp. on Commun.*, Queen's University, Kingston, Ont., Canada, May 1994.

19. F. R. Kschischang and G. B. Horn, "A Heuristic for Ordering a Linear Block Code to Minimize Trellis State Complexity," in *Proc. 32nd Allerton Conf. on Comm., Control, and Computing*, 75–84, Monticello, Ill. September 1994.

20. H. T. Moorthy and S. Lin, "On the Labeling of Minimal Trellises for Linear Block Codes," in *Proc. Int. Symp. Inform. Theory and Its Applications*, Sydney, Australia, 1: 33–38, November 1994.

21. A. Lafourcade and A. Vardy, "Asymptotically Good Codes Have Infinite Trellis Complexity," *IEEE Trans. Inform. Theory*, 41: 555–59, 1995.

22. O. Ytrehus, "On the Trellis Complexity of Certain Binary Linear Block Codes," *IEEE Trans. Inform. Theory*, 40: 559–60, 1995.

23. Y. Berger and Y. Be'ery, "Trellis-Oriented Decomposition and Trellis Complexity of Composite Length Cyclic Codes," *IEEE Trans. Inform. Theory*, 41: 1185–91, 1995.

24. F. R. Kschischang and V. Sorokine, "On the Trellis Structure of Block Codes," *IEEE Trans. Inform. Theory*, 41: 1924–37, 1995.

25. A. Lafourcade and A. Vardy, "Lower Bounds on Trellis Complexity of Block Codes," *IEEE Trans. Inform. Theory*, 41: 1938–54, 1995.

26. C. C. Lu and S. H. Huang, "On Bit-Level Trellis Complexity of Reed–Muller Codes," *IEEE Trans. Inform. Theory*, 41: 2061–64, 1995.

27. A. Lafourcade and A. Vardy, "Optimal Sectionalization of a Trellis," *IEEE Trans. Inform. Theory*, 42: 689–703, 1996.

28. R. J. McEliece, "On the BCJR Trellis for Linear Block Codes," *IEEE Trans. Inform. Theory*, 42: 1072–92, 1996.

29. M. P. C. Fossorier and S. Lin, "Coset Codes Viewed as Terminated Convolutional Codes," *IEEE Trans. Comm.*, 44: 1096–106, September 1996.

30. A. B. Kiely, S. Dolinar, R. J. McEliece, L. Ekroot, and W. Lin, "Trellis Decoding Complexity of Linear Block Codes," *IEEE Trans. Inform. Theory*, 42: 1687–97, 1996.

31. Y. Berger and Y. Be'ery, "The Twisted Squaring Construction, Trellis Complexity and Generalized Weights of BCH and QR Codes," *IEEE Trans. Inform. Theory*, 42: 1817–27, 1996.

32. F. R. Kschischang, "The Trellis Structure of Maximal Fixed-Cost Codes," *IEEE Trans. Inform. Theory*, 42: 1828–38, 1996.

33. A. Vardy and F. R. Kschischang, "Proof of a Conjecture of McEliece Regarding the Expansion Index of the Minimal Trellis," *IEEE Trans. Inform. Theory*, 42: 2027–33, 1996.

34. V. R. Sidorenko, G. Markarian, and B. Honary, "Minimal Trellis Design for Linear Block Codes Based on the Shannon Product," *IEEE Trans. Inform. Theory*, 42: 2048–53, 1996.

35. G. B. Horn and F. R. Kschischang, "On the Intractability of Permuting a Block Code to Minimize Trellis Complexity," *IEEE Trans. Inform. Theory*, 42: 2042–48, 1996.

36. T. Komura, M. Oka, T. Fujiwara, T. Onoye, T. Kasami, and S. Lin, "VLSI Architecture of a Recursive Maximum Likelihood Decoding Algorithm for a (64, 35) Subcode of the (64, 42) Reed–Muller Code," *Proc. Int. Symp. Inform. Theory and Its Applications*, Victoria, Canada, 709–12, September 1996.

37. H. T. Moorthy, "Efficient Near-Optimum Decoding Algorithms and Trellis Structure for Linear Block Codes," Ph.D. diss. University of Hawaii at Manoa, November 1996.

38. H. T. Moorthy, S. Lin, and G. T. Uehara, "Good Trellises for IC Implementation of Viterbi Decoders for Linear Block Codes," *IEEE Trans. Comm.*, 45: 52–63, 1997.

39. H. T. Moorthy, S. Lin and T. Kasami, "Soft-Decision Decoding of Binary Linear Block Codes Based on an Iterative Search Algorithm," *IEEE Trans. Inform. Theory*, 43: 1030–40, May 1997.

40. T. Kasami, T. Koumoto, T. Fujiwara, H. Yamamoto, Y. Desaki, and S. Lin, "Low Weight Subtrellises for Binary Linear Block Codes and Their Applications," *IEICE Trans. Fundamentals*, E80-A (11): 2095–103, November 1997.

41. B. Honary and G. Markarian, *Trellis Decoding of Block Codes: A Practical Approach*, Kluwer Academic, Boston, Mass., 1997.

42. R. H. Morelos-Zaragoza, T. Fujiwara, T. Kasami, and S. Lin, "Constructions of Generalized Concatenated Codes and Their Trellis-Based Decoding Complexity," *IEEE Trans. Inform. Theory*, 45; 725–731, March 1999.