

Example 8.4: Again, we consider the $(2,1,2)$ convolutional code given in Example 8.2. The trellis diagram corresponds to a message sequence of 5 bits long. The trellis has a **depth** of 7 as shown in Figure 8.7.

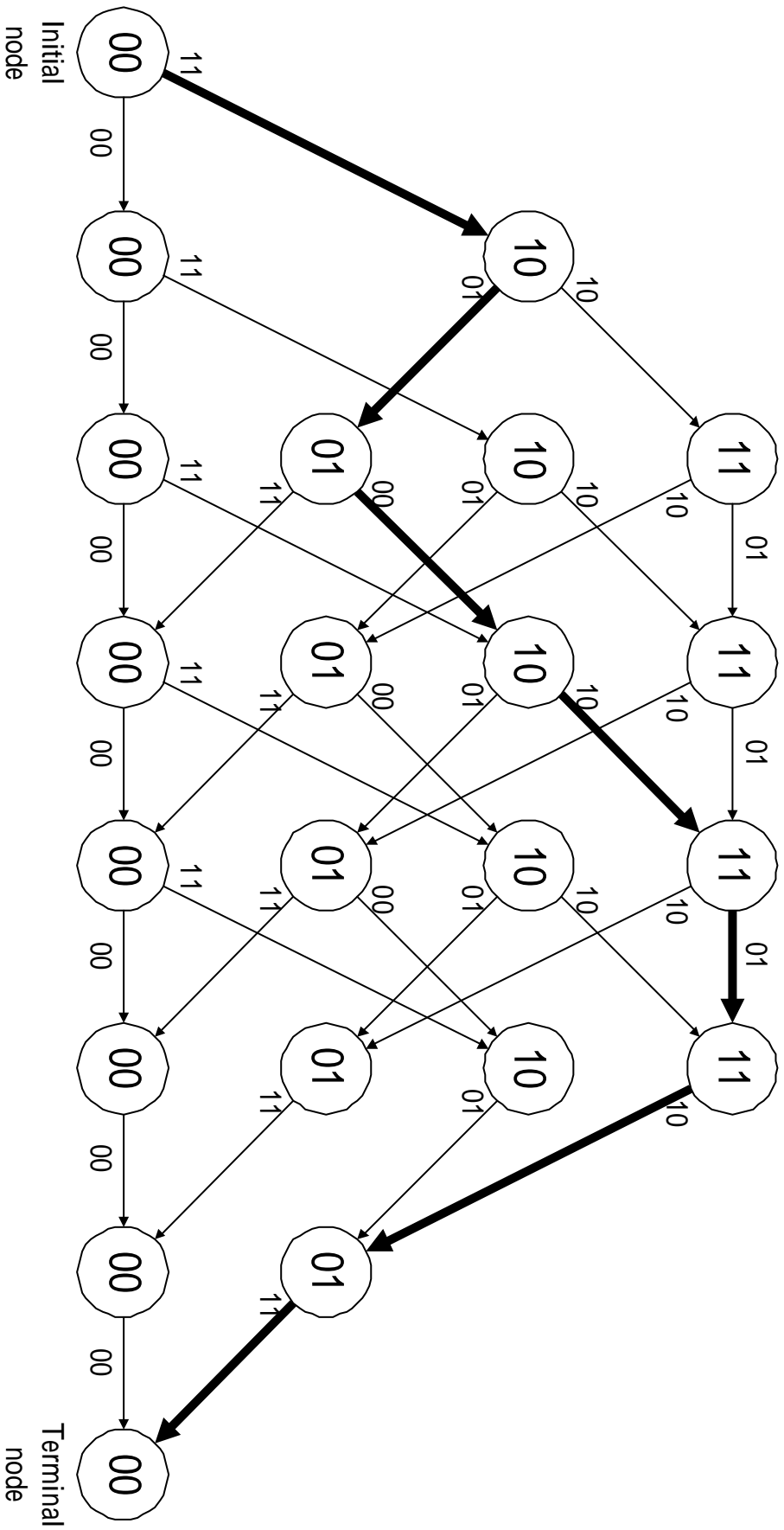


Figure 8.7 A terminated trellis diagram with $L = 7$

Further Structure

- The trellis diagram of a convolutional code consists of nodes (representing the state of the encoder) and branches (representing state transitions).
- The starting node of a trellis diagram is called the **initial** node.
- A sequence of consecutive branches connecting the initial node to a node in the trellis is called a **path**.
- The number of branches on a path is called the **length** of the path.
- A node, which is connected to the initial node by a path of length l , is called an l -th order node.

- For $l > m$, there is exactly one path of length l **entering** a l -th order node.
- For $l > m$, there are exactly 2^{l-m} paths of length l entering a l -th order node.
- There are a total of $2l$ paths of length l .
- Suppose two paths in the trellis branch off at a certain node. They will remerge at a later node if there are m consecutive branches on both paths corresponding to the same m message bits. This is because these m message bits take both paths to the same state.

Example 8.5: Consider the (3,2,1) convolutional code given in Example 8.1. Each state is of the form,

$$(c_{l-1}^{(1)}, c_{l-1}^{(2)}).$$

- There are four possible state:

$$S_0 = (00), \quad S_1 = (10)$$

$$S_2 = (01), \quad S_3 = (11)$$

- Since $k = 2$, there are four possible input message blocks at any time instant.
- Therefore, there are four possible transitions from a given state.
- The state diagram of the code is shown in Figure 8.8.

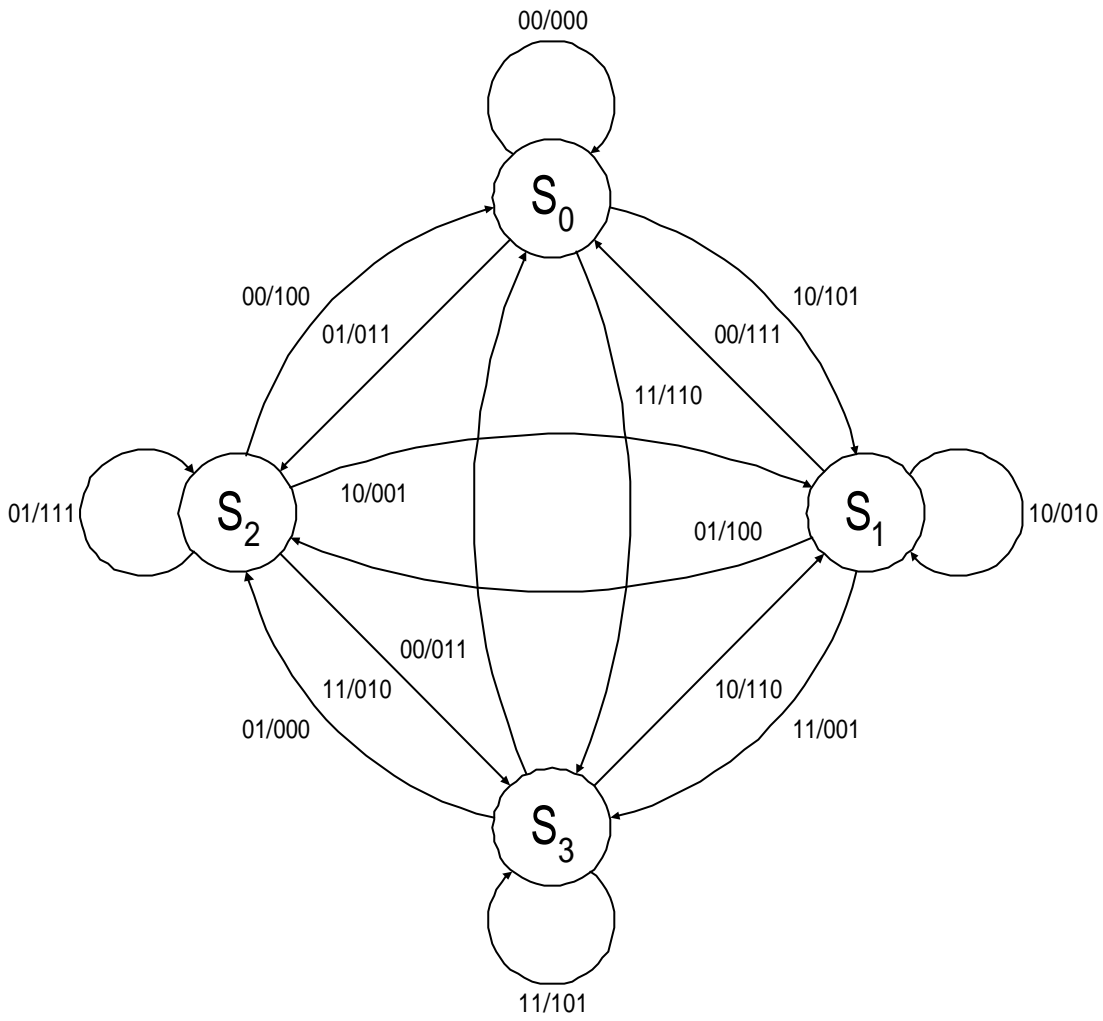


Figure 8.8 State diagram for a (3,2,1) code

5. Minimum Free Distance

- The most important distance measure for convolutional codes is the **minimum free** distance, denoted d_{free} .
- The minimum free distance of a convolutional code is simply the minimum Hamming distance between any two code sequences in the code.
- It is also the minimum weight of all the code sequences, which are produced by the nonzero message sequences.
- Since convolutional codes are linear, we can use the all-zero path $\bar{0}$ as our reference for studying the distance structure of convolutional codes without loss of generality.
- Let Q_l denote the set of those paths in the trellis diagram which diverge from the all-zero path $\bar{0}$ and then remerge with $\bar{0}$ for the first time at the same l -th order node.

- Let \bar{z} be a path in Q_1 . The part of \bar{z} which does not lie on $\bar{0}$ is called the **side-loop** of \bar{z} .
- A side-loop is at least $m+1$ branches long.
- The Hamming distance between \bar{z} and the all-zero path $\bar{0}$ is

$$d(\bar{z}, \bar{0}) = w(z) \\ = \text{the Hamming weight of the side-loop of } \bar{z} \quad (13)$$

- The minimum free distance of a convolutional code can be computed from the following expression:

$$d_{free} = \min_{l > m} \{ \min w(\bar{z}) : \bar{z} \in Q_1 \}. \quad (14)$$

- The minimum free distance of the (2,1,2) convolutional code given in Example 1 is 5, i.e., $d_{free}=5$.

6. Weight Distribution

- Like block codes, the weight distribution of a convolutional code determines its error performance.
- This weight distribution information can be obtained by modifying the state diagram of a code.
- The modification is done by splitting the all-zero state S_0 into two states, called the **initial** and **final** states. Then remove the self-loop around S_0 .
- Each path connecting the initial state and the final state represents a code sequence which diverges from and remerges with the all-zero state S_0 exactly once.
- In trellis diagram, this path represents a sequence which diverges from the all-zero path and then remerges for the first time at a later node.

- Let A_i denote the code sequences of weight i in the trellis which diverge from the all-zero path at the same node and then remerge for the first time at later nodes.
- A_i actually is **equal** to the number of paths of weight i in the modified state diagram which connect the initial and final state.

- The polynomial

$$T(X) = \sum_i A_i X^i \quad (15)$$

is called the **weight distribution function** of the code.

- $T(X)$ can be obtained by considering the modified state diagram as a signal flow diagram and then applying **Mason's gain formula**.
- It turns out that $T(X)$ is the **generating function** of the modified state diagram.

Generating Function

- Each branch in the modified state diagram is labeled with a branch gain X^i where i denote the weight of the branch.
- The gain X^l of a path is the product of branch gains along the path, i.e.,

$$X^l = X^{i_1} \cdot X^{i_2} \dots X^{i_j}$$

where $l = i_1 + i_2 + \dots + i_j$. Hence l is simply the weight of the path.

- A path connecting the initial state to the final state which does not go through any state twice is called a **forward path**.
- Let F_i represent the gain of the i -th forward path.

- A closed path starting at any state and returning to that state without going through any other state twice is called a **loop**.
- Let C_i denote the gain of the i -th loop.
- Two loops are said to be **non-touching** if they do not have any states in common.
- Let $\{i\}$ denote the set of all loops.
- Let $\{i,j\}$ denote the set of all pairs of non-touching loops.
- Let $\{i,j,k\}$ denote the set of triplets of non-touching loops, and so on.
- Define

$$\Delta = 1 - \sum_{\{i\}} C_i + \sum_{\{i,j\}} C_i C_j - \sum_{\{i,j,k\}} C_i C_j C_k + \dots \quad (16)$$

- Let G_i be the graph obtained by removing all the states on the i -th forward path and all the branches connecting to these states.
- Let Δ_i be the quantity defined by (16) but associated with G_i .
- By Mason's formula, we have

$$T(X) = \frac{\sum_i F_i \Delta_i}{\Delta} \quad (17)$$

Example 8.6: Again we consider the (2,1,2) code given in Example 8.2. The state diagram of this code is shown in Figure 8.5. The modified state diagram (signal-flow graph) G is shown in Figure 7.9. Each branch is labeled with its gain.

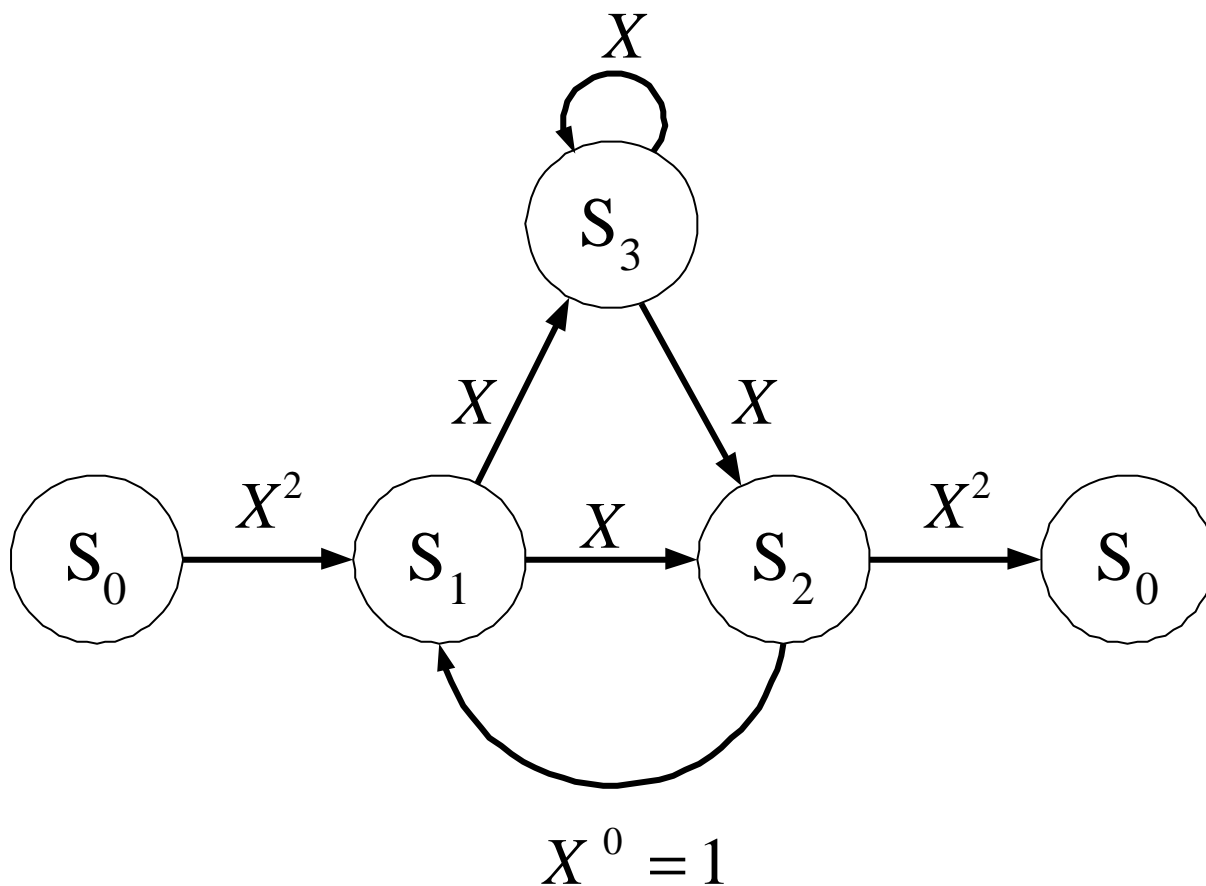


Figure 8.9 Modified state diagram for a (2,1,2) code

- There are 3 loops:
 - (1) S_3S_3 with gain $C_1=X$,
 - (2) $S_1S_2S_1$ with gain $C_2=X$,
 - (3) $S_1S_3S_2S_1$ with gain $C_3=X^2$.

- Pairs of non-touching loops:
 - $\{(1,2)\}$ with $C_1C_2=X^2$

- No triple non-touching loops.

- Hence,

$$\begin{aligned}
 \Delta &= 1 - (C_1 + C_2 + C_3) + C_1C_2 \\
 &= 1 - 2X - X^2 + X^2 \\
 &= 1 - 2X
 \end{aligned}$$

- The forward paths of G are:
 - (1) $S_0S_1S_2S_0$ with gain $F_1=X^5$.
 - (2) $S_0S_1S_3S_2S_0$ with gain $F_2=X^6$.

- The graph G_i obtained by removing the states on the forward path-1 is shown in Figure 8.10.

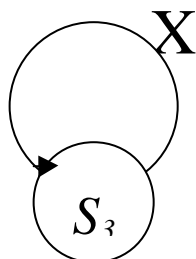


Figure 8.10 Graph G_1

- We find $\Delta_1 = 1 - X$.
- The graph G_2 obtained by removing the states on the forward path-2 is empty.
- Hence $\Delta_2 = 1$.
- Consequently, we have

$$\begin{aligned}
T(X) &= \frac{F_1\Delta_1 + F_2\Delta_2}{\Delta} \\
&= \frac{X^5(1-X) + X^6 \cdot 1}{1-2X} \\
&= \frac{X^5}{1-2X}
\end{aligned}$$

- Expanding $T(X)$ in series,

$$T(X) = X^5 + 2X^6 + 4X^7 + \dots + 2^k X^{k+5} + \dots$$

- Note that $A_5=1$. That is, there is only one path of weight 5 in the trellis which diverges from the all-zero path $\bar{0}$ at a fixed node and then remerges for the first time at a later node.
- Hence $d_{free}=5$.
- $A_6=2$, $A_7=4$ and $A_{k+5}=2^k$.

Remarks

- The weight distribution function $T(X)$ gives the error performance of a code.
- For a code of large memory order m , $T(X)$ is very hard to compute. The computation complexity grows exponentially with m .
- In general, d_{free} is used to measure the error performance of a code.

Detail Weight Structure

- More information about the weight structure of a code can be obtained by using the above procedure.
- Augment the modified state diagram by labeling each branch corresponding to a **nonzero** message bit with Y , and labeling every branch with Z .
- As a result of this augmentation, the generation function is then

$$T(X, Y, Z) = \sum_{i,j,l} A_{i,j,l} X^i Y^j Z^l \quad (18)$$

- The coefficients $A_{i,j,l}$ denote the number of paths of weight i , which diverge and remerge with the all-zero path (exactly once) after l branches, whose corresponding message sequences have weight j (j nonzero message digits).

Example 8.7: Consider the (2,1,2) code given in Example 1. Its modified state diagram with augmented branch labels is shown in Figure 10. Following the same steps as in Example 5, we find the augmented generating function as follows:

$$\begin{aligned}
 T(X, Y, Z) &= X^5YZ^3 / (1 - XYZ - XYZ^2) \\
 &= X^5YZ^3 + X^6(Y^2Z^4 + Y^2Z^5) \\
 &\quad + X^7(Y^3Z^5 + 2Y^3Z^6 + Y^3Z^7) \\
 &\quad + \dots
 \end{aligned}$$

- We see that there is a term $2X^7Y^3Z^6$.
- This term implies that there are two paths of weight 7, which diverge and remerge with the all-zero path after 6 branches, whose corresponding message sequences have 3 nonzero message bits.

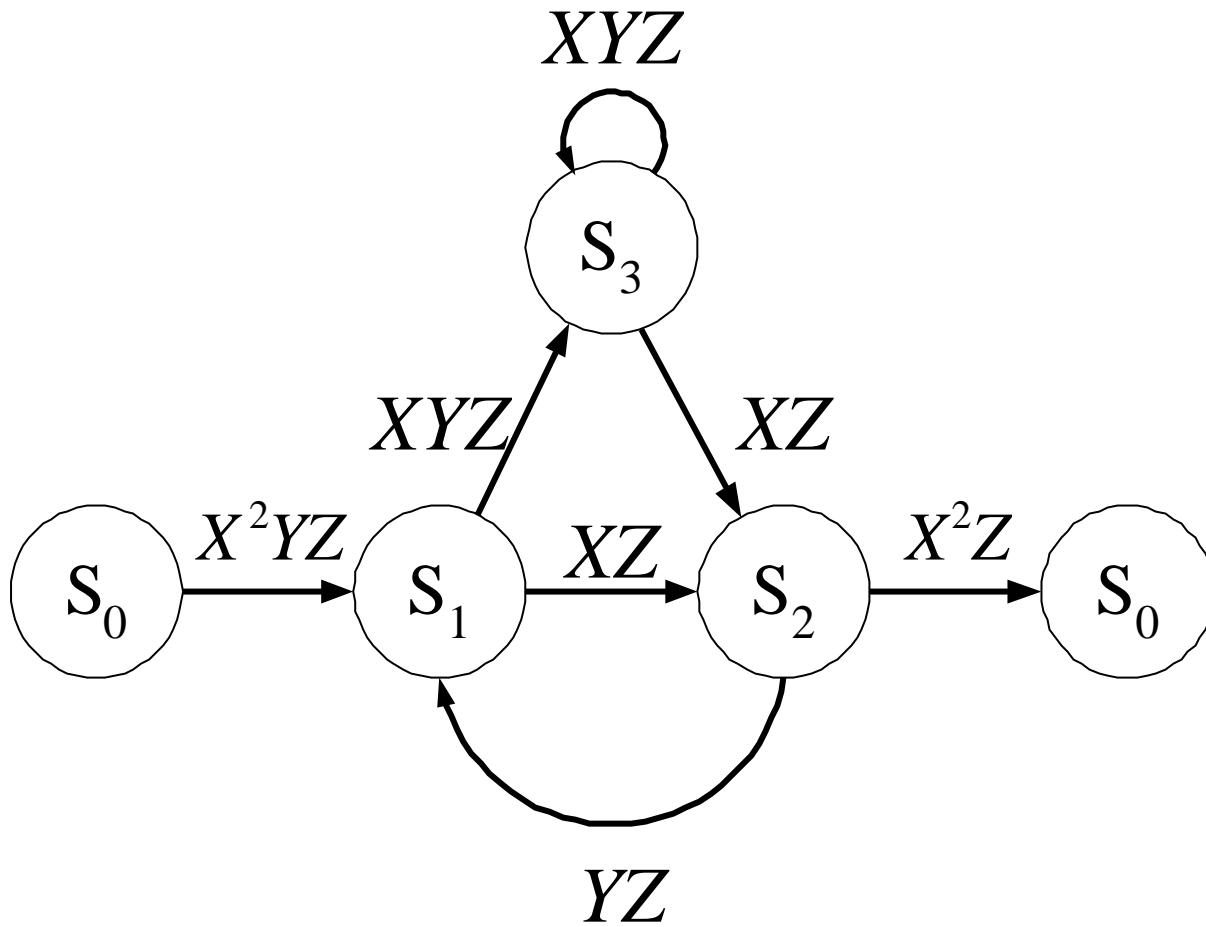


Figure 8.11 Augmented state diagram for a (2,1,2) code