For masking construction, we can use $\mathbb{H}_{EG,GA}(\rho)$ as the base matrix. This shortening gives more flexibility for constructing codes with various lengths and rates. For example, suppose we choose $EG(2, 2^7)$ for code construction using masking. If we choose $\gamma = 10$ and $\rho = 80$, the base matrix $\mathbb{H}_{EG,GA}(80)$ is then a 10×80 array of 128×128 permutation matrices. Masking this matrix with a 10×80 matrix $\mathbb{Z}$ with column and row weights of 4 and 32, respectively, we obtain a 1280×10240 masked matrix $\mathbb{M}$. The masking matrix $\mathbb{Z}$ consists of eight circulants that are constructed by using eight distinct primitive 10-tuples over $GF(2)$ and cyclically shifting each of them 10 times. The null space of $\mathbb{M}$ gives a (10240, 8961) LDPC code with rate 0.875 whose error performance is shown in Figure 17.32. At a BER of 10^{-6}, it performs only 0.9 dB from the Shannon limit.

Masking is a very powerful techniques for constructing both regular and irregular LDPC codes.

17.12 CONSTRUCTION OF QUASI-CYCLIC CODES BY CIRCULANT DECOMPOSITION

Consider a $q \times q$ circulant $\mathbb{G}$ over $GF(2)$ with column and row weights δ. Because column and row weights of a circulant are the same, for simplicity, we say that $\mathbb{G}$ has weight δ. For $1 \leq t \leq \delta$, let $w_1, w_2, \cdots, w_t$ be a set of positive integers such that $1 \leq w_1, w_2, \cdots, w_t \leq \delta$, and $w_1 + w_2 + \cdots + w_t = \delta$. Then, we can decompose $\mathbb{G}$ into t $q \times q$ circulants with weights $w_1, w_2, \cdots, w_t$, respectively. Let $\mathbf{g}_1$ be the first column of $\mathbb{G}$. We split $\mathbf{g}_1$ into t columns of the same length q, denoted by $\mathbf{g}_1^{(1)}$, $\mathbf{g}_1^{(2)}, \cdots, \mathbf{g}_1^{(t)}$, such that the first w_1 1-components of $\mathbf{g}_1$ are put in $\mathbf{g}_1^{(1)}$, the next w_2 1-components of $\mathbf{g}_1$ are put in $\mathbf{g}_1^{(2)}, \cdots$, and the last w_t 1-components of $\mathbf{g}_1$ are put in $\mathbf{g}_1^{(t)}$. For each new column $\mathbf{g}_1^{(i)}$, we form a $q \times q$ circulant $\mathbb{G}_i$ by cyclically shifting $\mathbf{g}_1^{(i)}$ downward q times. This results in t $q \times q$ circulants, $\mathbb{G}_1, \mathbb{G}_2, \cdots, \mathbb{G}_t$, with weights $w_1, w_2, \cdots, w_t$, respectively. These circulants are called the *descendants* of $\mathbb{G}$. Such a decomposition of $\mathbb{G}$ is called *column decomposition* of $\mathbb{G}$. Column decomposition of $\mathbb{G}$ results in a $q \times tq$ matrix,

$$\mathbb{H} = [\mathbb{G}_1 \mathbb{G}_2 \cdots \mathbb{G}_t], \tag{17.73}$$

which is a row of t $q \times q$ circulants. If $w_1 = w_2 = \cdots = w_t = w$, then $\mathbb{H}$ is a regular matrix with constant column weight w and constant row weight tw. If $t = \delta$ and $w_1 = w_2 = \cdots = w(\delta) = 1$, then each descendant circulant $\mathbb{G}_i$ of $\mathbb{G}$ is a permutation matrix, and $\mathbb{H}$ is a row of δ permutation matrices. The parameter t is called the *column splitting factor*. Figure 17.33 shows a column decomposition of an 8×8 circulant of weight 3 into two descendants with weights 2 and 1, respectively.

Similarly, we can decompose G into descendants by splitting its first row into multiple rows and cyclically shifting each new row to the right q times. Let $1 \leq c \leq max\{w_i : 1 \leq i \leq t\}$. For $1 \leq i \leq t$, let $w_{i,1}, w_{i,2}, \cdots, w_{i,c}$ be a set of nonnegative integers such that $0 \leq w_{i,1}, w_{i,2}, \cdots, w_{i,c} \leq w_i$, and $w_{i,1}+w_{i,2}+\cdots+w_{i,c} = w_i$. Each descendant $\mathbb{G}_i$ in $\mathbb{H}$ of (17.73) can be decomposed into c descendant circulants with weights $w_{i,1}, w_{i,2}, \cdots, w_{i,c}$, respectively. This is done by splitting the first row $\mathbf{g}_{i,1}$ of $\mathbb{G}_i$ into c rows of the same length q, denoted by $\mathbf{g}_{i,1}^{(1)}, \mathbf{g}_{i,1}^{(2)}, \cdots, \mathbf{g}_{i,1}^{(c)}$, where $\mathbf{g}_{i,1}^{(1)}$ contains the first $w_{i,1}$ 1-components of $\mathbf{g}_{i,1}$, $\mathbf{g}_{i,1}^{(2)}$ contains the second $w_{i,2}$ 1-components of $\mathbf{g}_{i,1}, \cdots$, and $\mathbf{g}_{i,1}^{(c)}$ contains the last $w_{i,c}$ 1-components of $\mathbf{g}_{i,1}$. Cyclically shifting each

$$\begin{bmatrix} 1&0&0&0&0&1&1&0\\ 0&1&0&0&0&0&1&1\\ 1&0&1&0&0&0&0&1\\ 1&1&0&1&0&0&0&0\\ 0&1&1&0&1&0&0&0\\ 0&0&1&1&0&1&0&0\\ 0&0&0&1&1&0&1&0\\ 0&0&0&0&1&1&0&1 \end{bmatrix} \longrightarrow \begin{bmatrix} 1&0&0&0&0&0&1&0\\ 0&1&0&0&0&0&0&1\\ 1&0&1&0&0&0&0&0\\ 0&1&0&1&0&0&0&0\\ 0&0&1&0&1&0&0&0\\ 0&0&0&1&0&1&0&0\\ 0&0&0&0&1&0&1&0\\ 0&0&0&0&0&1&0&1 \end{bmatrix} \begin{bmatrix} 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1\\ 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0 \end{bmatrix}$$

FIGURE 17.33: A column decomposition of a circulant of weight 3.

new row $\mathbf{g}_{i,1}^{(k)}$ to the right q times, we obtain c circulants $\mathbb{G}_i^{(1)}, \mathbb{G}_i^{(2)}, \cdots, \mathbb{G}_i^{(c)}$ that are descendants of $\mathbb{G}_i$. This decomposition is referred to as *row decomposition*. Row decomposition of $\mathbb{G}_i$ results in the following $cq \times q$ matrix:

$$\mathbb{D}_i = \begin{bmatrix} \mathbb{G}_i^{(1)} \\ \mathbb{G}_i^{(2)} \\ \vdots \\ \mathbb{G}_i^{(c)} \end{bmatrix}, \tag{17.74}$$

which is a column of c $q \times q$ circulants. We call $\mathbb{D}_i$ the row decomposition of $\mathbb{G}_i$ and c the *row splitting factor*. In row splitting, we allow $w_{i,k} = 0$. If $w_{i,k} = 0$, $\mathbb{G}_i^{(k)}$ is a $q \times q$ zero matrix, regarded as a circulant.

If each circulant $\mathbb{G}_i$ in $\mathbb{H}$ of (17.73) is replaced by its row decomposition $\mathbb{D}_i$, we obtain the following $c \times t$ array of circulants:

$$\mathbb{D} = \begin{bmatrix} \mathbb{D}_1 \, \mathbb{D}_2 \cdots \mathbb{D}_t \end{bmatrix} = \begin{bmatrix} \mathbb{G}_1^{(1)} & \mathbb{G}_2^{(1)} & \cdots & \mathbb{G}_t^{(1)} \\ \mathbb{G}_1^{(2)} & \mathbb{G}_2^{(2)} & \cdots & \mathbb{G}_t^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{G}_1^{(c)} & \mathbb{G}_2^{(c)} & \cdots & \mathbb{G}_t^{(c)} \end{bmatrix}, \tag{17.75}$$

which is a $cq \times tq$ matrix. If each $\mathbb{G}_i^{(k)}$ has weight 1, then $\mathbb{D}$ is an array of permutation matrices. The null space of $\mathbb{D}$ gives a code of length $n = tq$ that can be put in quasi-cyclic form (see Section 5.12).

If $\mathbb{G}$ is a sparse matrix, $\mathbb{D}$ is also a sparse matrix with smaller density than $\mathbb{G}$. If no two rows (or two columns) in $\mathbb{G}$ have more than one 1-component in common, then no two rows (or two columns) in $\mathbb{D}$ have more than one 1-component in common. In this case, the null space of $\mathbb{D}$ gives a quasi-cyclic LDPC code whose Tanner graph is free of cycles of length 4. If all the circulants in $\mathbb{D}$ have the same weight, then the code is a regular quasi-cyclic LDPC code.

As shown in Section 8.5 and Sections 17.4, 17.5, and 17.7, sparse circulants can be constructed from the incidence vectors of lines in either a Euclidean geometry or a projective geometry. Furthermore, no two rows (or two columns) either from the same or from two different circulants have more than one 1-component in common. Consequently, quasi-cyclic LDPC codes can be constructed by decomposing one or a group of these geometry circulants [54, 55].

The incidence vectors of all the lines in $\mathrm{EG}(m, 2^s)$ not passing through the origin can be partitioned into

$$K = (2^{(m-1)s} - 1)/(2^s - 1)$$

cyclic classes, $\mathcal{Q}_1, \mathcal{Q}_2, \cdots, \mathcal{Q}_K$. Each cyclic class $\mathcal{Q}_i$ consists of $2^{ms} - 1$ incidence vectors and can be obtained by cyclically shifting any vector in the class $2^{ms} - 1$ times. Hence, for each cyclic class $\mathcal{Q}_i$, we can form a $(2^{ms} - 1) \times (2^{ms} - 1)$ circulant $\mathbb{G}_i$ using any vector in $\mathcal{Q}_i$ as the first row and then cyclically shifting it $2^{ms} - 2$ times. The weight of $\mathbb{G}_i$ is 2^s. Consequently, we obtain a class of K circulants,

$$\mathcal{G} = \{\mathbb{G}_1, \mathbb{G}_2, \cdots, \mathbb{G}_K\}. \tag{17.76}$$

In code construction, we can take a subset of k circulants with $1 \leq k \leq K$, say $\mathbb{G}_1, \mathbb{G}_2, \cdots, \mathbb{G}_k$, from $\mathcal{G}$ and arrange them in a row,

$$\mathbb{H} = [\mathbb{G}_1 \mathbb{G}_2 \cdots \mathbb{G}_k].$$

We choose column and row splitting factors t and c, respectively, and decompose each circulant $\mathbb{G}_i$ in $\mathbb{H}$ into a $c \times t$ array $\mathbb{D}_i$ of descendants. Replacing each $\mathbb{G}_i$ in $\mathbb{H}$ by its array decomposition $\mathbb{D}_i$, we obtain a $c \times kt$ array $\mathbb{D}$ of descendant circulants. $\mathbb{D}$ is a $c(2^{ms} - 1) \times kt(2^{ms} - 1)$ matrix. The null space of $\mathbb{D}$ gives a quasi-cyclic LDPC code C whose Tanner graph does not contain cycles of length 4 and hence the girth of its Tanner graph is at least 6. If $\mathbb{D}$ has constant column and constant row weights, then C is a regular LDPC code.

EXAMPLE 17.27

Consider the three-dimensional Euclidean geometry $\mathrm{EG}(3, 2^3)$ over $GF(2^3)$. This geometry consists of 511 nonorigin points and 4599 lines not passing through the origin. The incidence vectors of the lines not passing through the origin form nine cyclic classes. From these cyclic classes we can form a class of nine 511×511 circulants of weight 8,

$$\mathcal{G} = \{\mathbb{G}_1, \mathbb{G}_2, \cdots, \mathbb{G}_9\}.$$

Suppose we take eight circulants from $\mathcal{G}$ and arrange them in a row to form the matrix

$$\mathbb{H} = [\mathbb{G}_1\ \mathbb{G}_2\ \mathbb{G}_3\ \mathbb{G}_4\ \mathbb{G}_5\ \mathbb{G}_6\ \mathbb{G}_7\ \mathbb{G}_8].$$

We choose column and row splitting factors $t = 2$ and $c = 4$, respectively, and decompose each circulant $\mathbb{G}_i$ into a 4×2 array $\mathbb{D}_i$ of eight 511×511 permutation matrices. Replacing each circulant $\mathbb{G}_i$ in $\mathbb{H}$ by its array decomposition $\mathbb{D}_i$, we obtain a 4×16 array $\mathbb{D}$ of 511×511 permutation matrices. $\mathbb{D}$ is a 2044×8176 matrix with column weight 4 and row weight 16. The null space of $\mathbb{D}$ gives an (8176, 6135) quasi-cyclic LDPC code with rate 0.75 and a minimum distance of at least 6. Its error performance with SPA decoding is depicted in Figure 17.34. It has a beautiful waterfall error performance all the way down to a BER of 10^{-10} without error floor. At a BER of 10^{-10}, it performs 1.4 dB from the Shannon limit.

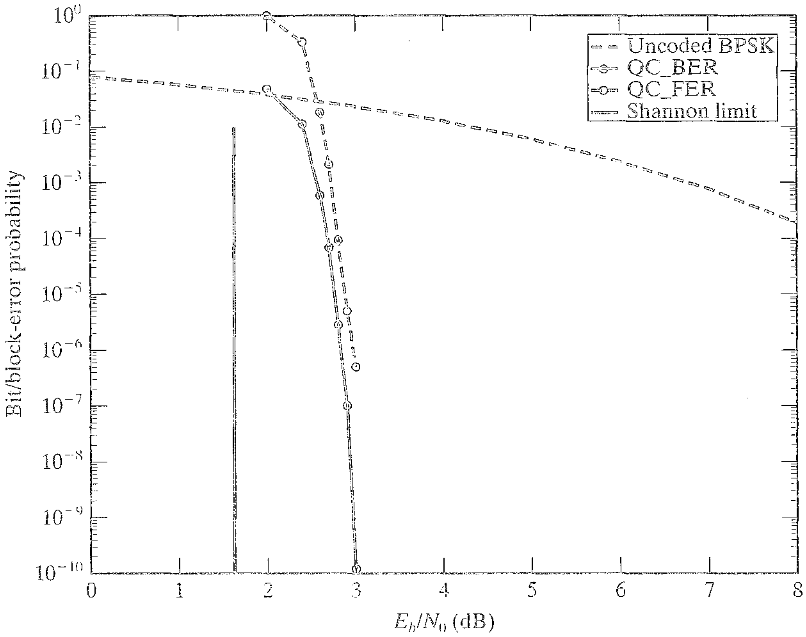

FIGURE 17.34: Error performance of the (8176, 6135) quasi-cyclic code given in Example 17.27.

Quasi-cyclic LDPC codes can also be constructed based on the incidence vectors of the lines in a projective geometry [17, 52, 55, 56]. Consider the m-dimensional projective geometry PG$(m, 2^s)$. There are two cases to be considered, even and odd m. For even m, the incidence vector of each line in PG$(m, 2^s)$ is primitive, and the incidence vectors of all the lines can be partitioned into

$$K_1 = (2^{ms} - 1)/(2^{2s} - 1)$$

cyclic classes. Each class consists of $(2^{(m+1)s} - 1)/(2^s - 1)$ incidence vectors that can be obtained by cyclically shifting any vector in the class. For odd $m \geq 3$, there are $(2^{(m+1)s} - 1)/(2^{2s} - 1)$ lines in PG$(m, 2^s)$ whose incidence vectors are not primitive, and the incidence vectors of all the other lines are primitive. The primitive incidence vectors can be partitioned into

$$K_2 = 2^s(2^{(m-1)s} - 1)/(2^{2s} - 1)$$

cyclic classes. Therefore, quasi-cyclic LDPC codes can be constructed based on the cyclic classes of incidence vectors of lines in a projective geometry.

EXAMPLE 17.28

Consider the three-dimensional projective geometry PG$(3, 2^3)$ over $GF(2^3)$. This geometry consists of 585 points and 4745 lines. Each line consists of 9 points. There are 65 lines whose incidence vectors are not primitive, and the incidence vectors of the other 4680 lines are primitive. The 4680 primitive incidence vectors can be partitioned into eight cyclic classes, with each class consisting of 585 vectors. Based on these eight cyclic classes of incidence vectors, we can form eight 585×585 circulants, $\mathbb{G}_1, \mathbb{G}_2, \cdots, \mathbb{G}_8$. Each circulant $\mathbb{G}_i$ has weight 9. Suppose we want to construct a quasi-cyclic LDPC code of length 9360 with rate 0.875. First, we decompose each circulant $\mathbb{G}_i$ by column decomposition into three descendant circulants, $\mathbb{G}_{i,1}$, $\mathbb{G}_{i,2}$, and $\mathbb{G}_{i,3}$, with weights 4, 4, and 1, respectively. Removing the descendant with weight 1, we obtain the 585×1170 matrix

$$\mathbb{D}_i = [\mathbb{G}_{i,1} \mathbb{G}_{i,2}],$$

which consists of two 511×511 descendant circulants of $\mathbb{G}_i$. We decompose each circulant of $\mathbb{D}_i$ by row decomposition into two descendants, each having weight 2. The result is the following 2×2 array of 511×511 circulants:

$$\mathbb{D}_i^* = \begin{bmatrix} \mathbb{G}_{i,1}^{(1)} & \mathbb{G}_{i,2}^{(1)} \\ \mathbb{G}_{i,1}^{(2)} & \mathbb{G}_{i,2}^{(2)} \end{bmatrix},$$

which is a 1170×1170 matrix with both column and row weight 4. We then form the following 1170×9360 matrix:

$$\mathbb{D} = [\mathbb{D}_1^* \, \mathbb{D}_2^* \, \mathbb{D}_3^* \, \mathbb{D}_4^* \, \mathbb{D}_5^* \, \mathbb{D}_6^* \, \mathbb{D}_7^* \, \mathbb{D}_8^*],$$

which has column and row weights 4 and 32, respectively. The null space of $\mathbb{D}$ gives a (9360, 8192) quasi-cyclic LDPC code with rate 0.875. The error performance of this code is shown in Figure 17.35. At a BER of 10^{-6}, it performs 0.95 dB from the Shannon limit.

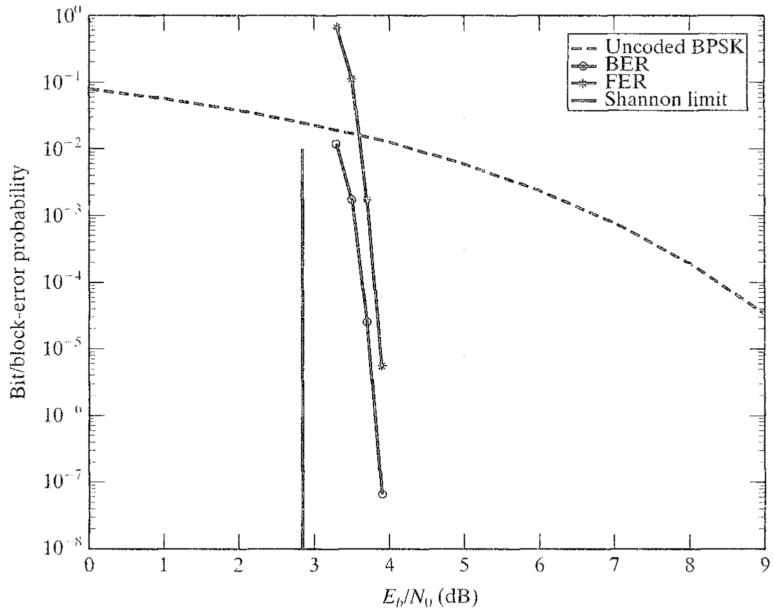

FIGURE 17.35: Error performance of the (9360, 8192) quasi-cyclic code given in Example 17.28.

17.13 CONSTRUCTION OF LDPC CODES BASED ON FINITE GEOMETRIES OVER $GF(p^s)$

So far, we have considered construction of LDPC codes based only on finite geometries over the field $GF(2^s)$; however, LDPC codes also can be constructed based on the lines and points of finite geometries over the field $GF(p^s)$, where p is a prime. The approaches to the code construction are exactly the same as those presented in previous sections. A finite geometry, Euclidean or projective, over $GF(p^s)$ has the same structural properties as a finite geometry over $GF(2^s)$. All the expressions for the number of points, the number of lines, the number of parallel bundles, and so on, derived for finite geometries over $GF(2^s)$ can be applied to finite geometries over $GF(p^s)$ simply by replacing 2 with p.

The m-dimensional Euclidean geometry EG(m, p^s) over $GF(p^s)$ consists of p^{ms} points, with each point an m-tuple over $GF(p^s)$. A line in EG(m, p^s) is simply a one-dimensional subspace or its coset of the vector space of all the m-tuples over $GF(p^s)$. Therefore, each line consists of p^s points. There are

$$p^{(m-1)s}(p^{ms}-1)/(p^s-1) \tag{17.77}$$

lines. These lines can be grouped into

$$K=(p^{ms}-1)/(p^s-1) \tag{17.78}$$

parallel bundles, each consisting of $p^{(m-1)s}$ lines parallel to each other. Each parallel bundle contains all the p^{ms} points of the geometry, and each point is on one and only one line in the parallel bundle. For each point, there are

$$(p^{ms}-1)/(p^s-1) \tag{17.79}$$

lines intersecting at it. The incidence vectors of all the lines not passing through the origin can be partitioned into

$$(p^{(m-1)s}-1)/(p^s-1) \tag{17.80}$$

cyclic classes, each consisting of $p^{ms}-1$ vectors.

All the constructions of LDPC codes based on the lines and points of EG(m, 2^s) can be directly applied to the construction of LDPC codes based on the lines and points of EG(m, p^s). We use the construction of EG-Gallager LDPC codes for illustration. Let $n=p^{ms}$. The incidence vector of a line $\mathcal{L}$ in EG(m, p^s) is an n-tuple over $GF(2)$,

$$\mathbb{v}_{\mathcal{L}}=(v_0, v_1, \cdots, v_{n-1}),$$

whose components correspond to the n points of EG(m, p^s) and $v_j=1$ if and only if v_j corresponds to a point on $\mathcal{L}$, and $v_j=0$ otherwise. Therefore, the weight of $\mathbb{v}_{\mathcal{L}}$ is p^s. For each parallel bundle P_i of lines in EG(m, p^s), we can form a $p^{(m-1)s}\times p^{ms}$ matrix $\mathbb{H}_i$ over $GF(2)$ whose rows are the incidence vectors of lines in P_i. It follows from the structural properties of a parallel bundle that the column and row weights of $\mathbb{H}_i$ are 1 and p^s, respectively. Let $1\le\gamma\le K$. We form a $\gamma p^{(m-1)s}\times p^{ms}$ matrix $\mathbb{H}_{EG,GA}$ over $GF(2)$ with γ submatrices arranged in a column as follows:

$$\mathbb{H}_{EG,GA}=\begin{bmatrix}\mathbb{H}_1\\ \mathbb{H}_2\\ \vdots\\ \mathbb{H}_\gamma\end{bmatrix}.$$

where each submatrix $\mathbb{H}_i$ is formed based on a parallel bundle P_i of lines in EG(m, p^s). The column and row weights of $\mathbb{H}_{EG,GA}$ are γ and p^s, respectively. The null space over $GF(2)$ of this binary matrix gives a Gallager LDPC code of length $n = p^{ms}$ and minimum distance of at least $\gamma + 1$ for odd γ and $\gamma + 2$ for even γ. Clearly, for $p = 2$, we obtain the class of EG-Gallager LDPC codes given in the previous sections.

EXAMPLE 17.29

Let $m = 2$, $s = 1$, and $p = 43$. The two-dimensional Euclidean geometry EG(2, 43) consists of 1849 points and 1892 lines. The 1892 lines can be grouped into 44 parallel bundles, each consisting of 43 lines parallel to each other. Each line consists of 43 points. Taking 4 and 6 parallel bundles, we can construct two EG-Gallager LDPC codes that are (1849, 1680) and (1849, 1596) codes with rates 0.9086 and 0.8632, respectively. With $\gamma = 4$ and 6, the lower bound $\gamma + 2$ on the minimum distance of an EG-Gallager code gives their minimum distances as at least 6 and 8, respectively. The bit-error performances of these two EG-Gallager LDPC codes with SPA decoding are shown in Figure 17.36 (assuming BPSK signaling). At a BER of 10^{-6}, the (1849, 1680) code achieves a 5.7-dB coding gain over the uncoded BPSK system and is 1.5 dB from the Shannon limit for rate 0.9086, and the (1849, 1596) code achieves a 6-dB coding gain over the uncoded BPSK system and is 2.1 dB from the Shannon limit for rate 0.8632.

EXAMPLE 17.30

Suppose it is desired to construct a rate-1/2 LDPC code with length around 6400 based on geometry decomposition and masking. To satisfy the length constraint, we must choose m, s, and p such that $p^{ms} \simeq 6400$. There are many possible choices of m, s, and p for which $p^{ms} \simeq 6400$. One such choice is $m = 2$, $s = 4$, and $p = 3$. With this choice, the geometry for decomposition is the three-dimensional Euclidean geometry EG$(2, 3^4)$. This geometry consists of 6561 points and 6642 lines, and each line consists of 81 points. The 6642 lines can be grouped into 82 parallel bundles, with each parallel bundle consisting of 81 parallel lines. We decompose this geometry based on a parallel bundle $P(2, 1)$ of lines. Then, there are 81 connecting parallel bundles of lines, $Q_1, Q_2, \cdots, Q_{81}$, with respect to $P(2, 1)$. The incidence matrix $\mathbb{A}_i$ of a connecting parallel bundle Q_i is an 81×6561 matrix consisting of eighty-one 81×81 permutation matrices. To achieve the desired rate 1/2 and length equal or close to 6400, we set $\rho = 80$ and $\gamma = 40$. With this choice of ρ and γ, we construct a base matrix $\mathbb{H}_{EG,GA}(80)$ based on (17.72). $\mathbb{H}_{EG,GA}$ is a 3240×6480 matrix with column and row weights 40 and 80, respectively. It is a 40×80 array of 81×81 permutation matrices. Next, we need to construct a 40×80 masking matrix $\mathbb{Z}$. Suppose we choose the column and row weights of $\mathbb{Z}$ to be 3 and 6, respectively. To construct $\mathbb{Z}$, we choose two distinct primitive 40-tuples over $GF(2)$. Cyclically shifting these two distinct primitive 40-tuples, we form two 40×40 circulants $\mathbb{G}_1$ and $\mathbb{G}_2$. Then, $\mathbb{Z} = [\mathbb{G}_1 \mathbb{G}_2]$. Because the weight of each circulant is 3, it is easy to construct $\mathbb{G}_1$ and $\mathbb{G}_2$ such that the Tanner graph of $\mathbb{Z}$ is free of cycles of length 4. Masking the base matrix $\mathbb{H}_{EG,GA}(80)$ with $\mathbb{Z}$, we obtain a 3240×6480 masked matrix

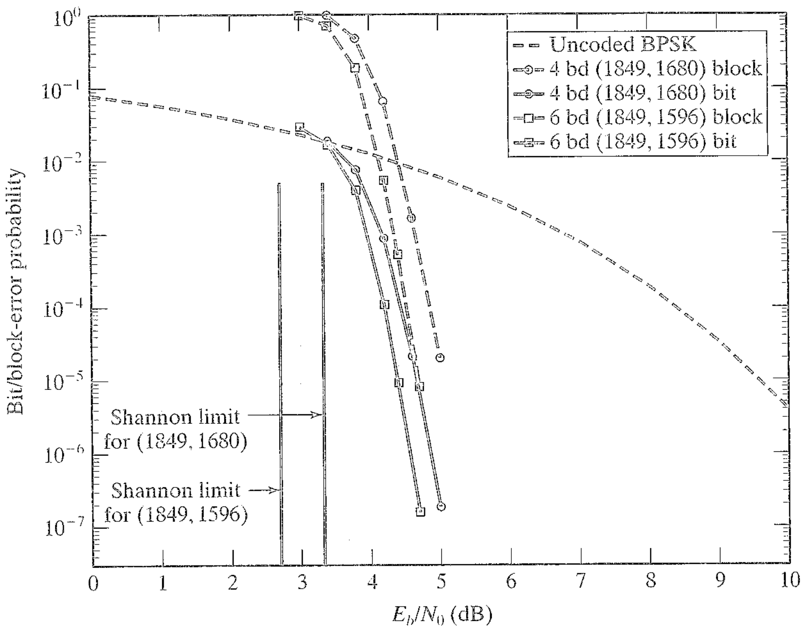

FIGURE 17.36: Error probabilities of two EG-Gallager codes of length 1849.

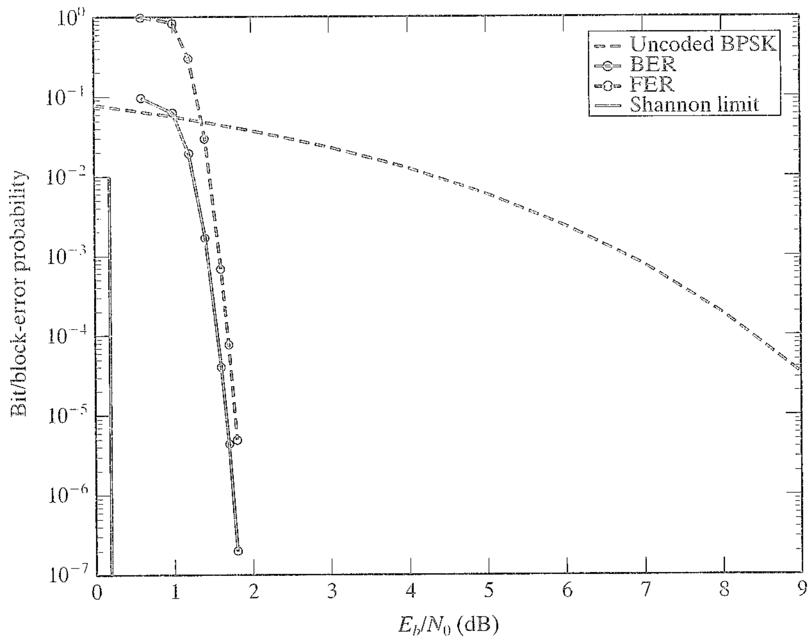

FIGURE 17.37: Bit- and block-error probabilities of a (6480, 3240) LDPC code.

$\mathbb{M} = \mathbb{Z} \otimes \mathbb{H}_{EG,GA}(80)$ with column and row weights 3 and 6, respectively. The null space of $\mathbb{M}$ gives a (3, 6) regular (6480, 3240) LDPC code with rate 1/2. The error performance of this code decoded with the SPA is shown in Figure 17.37. It has both good bit- and block-error performances. At a BER of 10^{-6}, it performs 1.5 dB from the Shannon limit.

Construction based on finite geometries over $GF(p^s)$ results in a very large class of finite-geometry LDPC codes containing all the finite-geometry LDPC codes constructed in the previous sections of this chapter as subclasses.

17.14 RANDOM LDPC CODES

In addition to being formed by the geometric construction presented in the previous sections, LDPC codes also can be constructed by computer search in a pseudorandom manner based on a set of guidelines satisfying the conditions given in Definition 17.1. This construction results in an ensemble of random codes that have been proved to contain good LDPC codes [10].

Suppose it is desired to construct an LDPC code of length n with rate k/n. To construct a parity-check matrix for this desired code, we need to choose an appropriate column weight γ and an appropriate number J of rows. It is clear that J must be at least equal to $n - k$, the number of parity-check symbols of the desired code. In computer construction, J is usually chosen to be equal to $n - k$. For $\mathbb{H}$ to have constant row weight ρ, the condition

$$\gamma \times n = \rho \times (n - k) \tag{17.81}$$

must hold. Otherwise, $\mathbb{H}$ cannot have constant row weight. In this case, we simply try to keep all the row weights close to ρ. If n is divisible by $n - k$, it follows from (17.81) that ρ is a multiple of γ; that is, $\rho = \gamma n/(n - k)$. For this case, we can construct a regular low-density parity-check matrix with column weight γ and row weight ρ. If n is not divisible by $n - k$, we divide $\gamma \times n$ by $n - k$ and obtain

$$\gamma \times n = \rho(n - k) + b, \tag{17.82}$$

where ρ and b are the quotient and remainder, respectively, with $0 < b < n - k$. The expression of (17.82) can be rearranged as follows:

$$\gamma \times n = (n - k - b)\rho + b(\rho + 1). \tag{17.83}$$

which suggests that we can construct a low-density parity-check matrix $\mathbb{H}$ with two row weights, ρ and $\rho + 1$, respectively. For convenience, $\mathbb{H}$ is constructed in such a way that its top b rows have weight $\rho + 1$, and its bottom $n - k - b$ rows have weight ρ.

The construction of $\mathbb{H}$ is carried out step by step. At each step, one column is added to a partially formed matrix. Each added column must satisfy certain constraints. For $1 \leq i \leq n$, at the ith step, a binary $(n - k)$-tuple of weight γ is chosen as a candidate column $\mathbb{h}_i$ and is added to the partial parity-check matrix

$$\mathbb{H}_{i-1} = [\mathbb{h}_1, \mathbb{h}_2, \cdots, \mathbb{h}_{i-1}] \tag{17.84}$$

obtained at the $(i-1)$th step to form the next partial parity-check matrix $\mathbb{H}_i$. For $i = 1$, $\mathbb{H}_0$ is simply the null matrix. To add column $\mathbb{h}_i$ to $\mathbb{H}_{i-1}$, the following constraints must be satisfied:

1. Choose $\mathbb{h}_i$ at random from the remaining binary $(n-k)$-tuples that are not being used in $\mathbb{H}_{i-1}$ and that were not rejected earlier.
2. Check whether $\mathbb{h}_i$ has more than one 1-component in common with any column in $\mathbb{H}_{i-1}$. If not, go to step 3; otherwise, reject $\mathbb{h}_i$ and go back to step 1 to choose another candidate column.
3. Add $\mathbb{h}_i$ to $\mathbb{H}_{i-1}$ to form a temporary partial parity-check matrix $\mathbb{H}_i$. Check the row weights of $\mathbb{H}_i$. If all the top b rows of $\mathbb{H}_i$ have weights less than or equal to $\rho+1$, and all the bottom $n-k-b$ rows of $\mathbb{H}_i$ have weights less than or equal to ρ, then permanently add $\mathbb{h}_i$ to $\mathbb{H}_{i-1}$ to form $\mathbb{H}_i$ and go to step 1 to continue the construction process. If any of the top b rows of $\mathbb{H}_i$ has weight exceeding $\rho+1$, or any of the bottom $n-k-b$ rows of $\mathbb{H}_i$ has weight exceeding ρ, reject $\mathbb{h}_i$ and go to step 1 to choose another candidate column.

The step-by-step construction process continues until a parity-check matrix $\mathbb{H}$ with n columns is formed. If $b = 0$, $\mathbb{H}$ is a regular matrix with row and column weights ρ and γ, respectively. If $b \neq 0$, then $\mathbb{H}$ has two row weights, $\rho+1$ and ρ. For a given n, k, and γ, it is possible that all the $(n-k)$-tuples are either used or rejected before $\mathbb{H}$ is formed. To reduce this possibility, we need to choose n, k, and γ such that the total number of binary $(n-k)$-tuples, $\binom{n-k}{\gamma}$, is much larger than the code length n, or we can relax the row weight constraints in step 3 to allow multiple row weights. This also reduces the probability that a chosen candidate column that satisfies the constraint in step 2 will be rejected in step 3. Of course, we can restart the construction process by choosing another sequence of candidate columns.

If the row rank of $\mathbb{H}$ is exactly $n-k$, the null space of $\mathbb{H}$ gives an (n, k) LDPC code with rate exactly k/n. If the rank of $\mathbb{H}$ is less than $n-k$, then the null space gives an (n, k') LDPC code with $k' > k$ and rate $k'/n > k/n$. We can readily show that the rate R of the constructed code is lower bounded as follows:

$$R \geq 1 - \frac{\gamma}{\rho}.$$

The constraint at step 2 of column selection ensures that the Tanner graph of the code does not contain any cycle of length 4. Therefore, the girth of the Tanner graph is at least 6. The foregoing construction is efficient only for small γ, usually 3 or 4. For large γ, to check the constraints at steps 2 and 3 can be computationally expensive. Because at step 1 of the construction a column is chosen at random from the remaining available binary $(n-k)$-tuples, the code constructed is not unique. The construction gives an ensemble of random LDPC codes. With this construction it is very hard to determine the minimum distance of the code constructed. For small γ, 3 or 4, the lower bound, $\gamma+1$, on the minimum distance can be very poor.

EXAMPLE 17.31

Suppose we want to construct a (504, 252) LDPC code with rate 1/2. We choose $\gamma = 3$. Because $n = 504$, and $n-k = 252$, $n/(n-k) = 2$, so we choose row weight

$\rho = 2 \times 3 = 6$. The number of binary 252-tuples with weight 3 is

$$\binom{252}{3} = 2635500,$$

which is much larger than the code length $n = 252$. Following the construction procedure, we obtain a regular 252×504 parity-check matrix $\mathbb{H}$ with column and row weights 3 and 6, respectively. The density of $\mathbb{H}$ is $6/504 = 0.012$. Its row rank is 252, and hence its null space gives a (504, 252) LDPC code with rate 1/2. Because γ is 3, the minimum distance of the code is at least 4. The true minimum distance of this code may be greater than 4. The bit-error performance of this code with SPA decoding is shown in Figure 17.29. We see that this code performs very close to the rate-1/2 (512, 256) EG-Gallager code given in Example 17.23.

Random construction results in a large ensemble of LDPC codes that contains finite-geometry LDPC codes as a subclass. Clearly, there must exist random LDPC codes that outperform finite-geometry LDPC codes in error performance, especially long random codes. Computer-generated random LDPC codes, in general, do not have the structural properties of the finite-geometry LDPC codes, such as cyclic or quasi-cyclic structure. Consequently, encoding of a random LDPC code in hardware is much more complex than encoding a finite-geometry LDPC code; that is, its encoding cannot be implemented with linear shift registers. Because computer-generated LDPC codes, in general, have relatively small column weight, the number of check-sums orthogonal on a code bit that can be formed is small. As a result, these codes perform poorly with one-step majority-logic decoding or bit-flipping decoding. Furthermore, computer-generated random LDPC codes with SPA decoding do not converge as fast as finite-geometry LDPC codes do. For finite-geometry LDPC codes with SPA decoding, usually 5 iterations give a performance only a fraction of a decibel from their performance with 100 iterations, as demonstrated in Figure 17.11 for the (4095, 3367) cyclic EG-LDPC code. With all these disadvantages, long random LDPC codes do perform very close to the Shannon limit. For example, very long random LDPC codes (10^7 bits long) have been constructed and shown to perform only a few thousandths of a decibel from the Shannon limit [14].

17.15 IRREGULAR LDPC CODES

An irregular LDPC code is defined by a parity-check matrix $\mathbb{H}$ with multiple column weights and multiple row weights. In terms of its Tanner graph, the variable nodes (code-bit vertices) have multiple degrees, and the check nodes (check-sum vertices) have multiple degrees. It has been shown that long random irregular codes perform arbitrarily close to the Shannon limit [8, 12–14]. Irregular LDPC codes are most commonly designed and constructed based on their Tanner graphs. One such approach is to design these codes in terms of the degree distributions of the variable and check nodes of their Tanner graphs [12, 13].

Consider the Tanner graph $\mathcal{G}$ of an irregular LDPC code with parity-check matrix $\mathbb{H}$. The variable nodes in $\mathcal{G}$ correspond to the columns of $\mathbb{H}$, and the check nodes of $\mathcal{G}$ correspond to the rows of $\mathbb{H}$. The degree of a node in $\mathcal{G}$ is defined as the number of edges connected to it. The degree of a variable node is simply equal to the weight of its corresponding column in $\mathbb{H}$, and the degree of a check node is simply equal to the weight of its corresponding row in $\mathbb{H}$. Let

$$\gamma(X) = \sum_{i=1}^{d_v} \gamma_i X^{i-1} \tag{17.85}$$

be the *degree distribution* of the variable nodes of $\mathcal{G}$, where γ_i denotes the fraction of variable nodes in $\mathcal{G}$ with degree i, and d_v denotes the maximum variable-node degree. Let

$$\rho(X) = \sum_{i=1}^{d_c} \rho_i X^{i-1} \tag{17.86}$$

be the degree distribution of the check nodes of $\mathcal{G}$, where ρ_i denotes the fraction of check nodes in $\mathcal{G}$ with degree i, and d_c denotes the maximum check-node degree.

In [12] and [13] it has been shown that the error performance of an irregular LDPC code depends on the variable- and check-node degree distributions of its Tanner graph. Shannon limit–approaching irregular LDPC codes can be designed by optimizing the two degree distributions via algorithms centered on the evolution of the probability densities of the messages passed between the two types of nodes in a belief propagation decoder; however, the design algorithm presented in [12] gives optimal degree distributions for a given code rate only when the code length approaches infinity, and the code graph is cycle free with an infinite number of decoding iterations. Tanner graphs for codes of finite lengths cannot be made cycle free; hence, the optimal degree distributions designed for infinite code length will not be optimal any more for codes of finite lengths. If we construct an irregular LDPC code of finite length based on the asymptotically optimal degree distributions, a high error floor may result, and the code may perform poorly in the low-bit error-rate range owing to the large number of degree-2 variable nodes in its Tanner graph. A large number of degree-2 variable nodes, in general, results in poor minimum distance. To overcome these problems, the following additional design rules are proposed in [12]: (1) the degree-2 variable nodes are made cycle free; (2) degree-2 variable nodes are made to correspond to the parity check bits of a codeword; and (3) the code graph is free of cycles of length 4. In order to improve the error floor performance, another constraint is proposed in [21] and [22]. This constraint requires that the number of degree-2 variable nodes must be smaller than the number of parity check bits of the code (or the number of columns of the parity check matrix $\mathbb{H}$ with weight 2 must be smaller than the number of rows of $\mathbb{H}$).

The requirement that the number of degree-2 variable nodes be smaller than the number of parity-check bits of the code to be designed can be met with a simple degree redistribution. Let n_d and R_d be the desired code length and rate, respectively. First, we design the asymptotically optimal degree distribution $\gamma(X)$ and $\rho(X)$ for the desired rate based on the evolution of probability densities. From the fraction of the degree-2 variable nodes given in $\gamma(X)$, we compute the number N_2 of degree-2 variable nodes using the desired code length n_d; that is, $N_2 = \gamma_2 \times n_d$. If N_2 is smaller than the number $n_d(1 - R_d)$ of the parity-check bits of the desired code, then $\gamma(X)$ and $\rho(X)$ are used for constructing the desired code. If N_2 is greater than $n_d(1 - R_d)$, then we convert the excess degree-2 variable nodes to variable nodes of next-higher degree, say, degree 3 if it exists. The result is a modified variable-node degree distribution $\gamma^*(X)$. In a Tanner graph, the sum of check-node degrees (equal to the total number of edges of the graph) is equal to the sum of variable-node degrees. If we modify the variable-node degree distribution $\gamma(X)$, we must change the check-node degree distribution $\rho(X)$ accordingly to make the sum of check-node degrees the same as the sum of variable-node degrees in the new variable-node degree distribution.

The next step in code construction is to construct a Tanner graph by connecting the variable nodes and check nodes with edges under the constraints given by the degree distributions. Because the selection of edges in the graph construction is not unique, edges are selected randomly. During the edge selection process, effort must be made to avoid having cycles among the degree-2 variable nodes and cycles of length 4 in the code graph. As a result, computer search is needed. Once a code graph is constructed, we form the corresponding parity-check matrix $\mathbb{H}$. Then, the column and row weight distributions of $\mathbb{H}$ are the same as the variable- and check-node degree distributions. The null space of $\mathbb{H}$ gives an irregular code of the desired length and rate. The described construction gives a random irregular LDPC code.

The masking technique presented in Section 17.11 can be used to simplify the construction of irregular LDPC codes based on the degree distributions of variable and check nodes of their Tanner graphs. For masking, the array $\mathbb{H}_{EG,GA}(\rho)$ of permutation matrices given by (17.72) is used as the base matrix. Suppose the degree distributions $\gamma(X)$ and $\rho(X)$ have been designed for a given code rate R_d and length n_d. Suppose geometry EG(m, p^s) is used for constructing the base matrix. We choose parameters γ and ρ such that $\rho p^{(m-1)s}$ is equal or close to the desired code length n_d, and $(\rho - \gamma)/\gamma$ is equal or close to the desired rate R_d. We construct a $\gamma \times \rho$ masking matrix $\mathbb{Z} = [z_{i,j}]$ with column and row weight distributions $\mathbb{c}(X)$ and $\mathbb{d}(X)$, respectively, identical to the degree distributions $\gamma(X)$ and $\rho(X)$ of the variable and check nodes, respectively, where $d_v \leq \gamma$ and $d_c \leq \rho$.

In constructing the masking matrix $\mathbb{Z}$, we put columns with weight 2 in the parity-check positions and make them cycle free among them. The cycle-free condition can easily be achieved by using a set of weight-2 columns obtained by downward shifting two consecutive 1's from the top of $\mathbb{Z}$ until the second 1 reaches the bottom of $\mathbb{Z}$ as shown:

$$\mathbb{Z} = \left[\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & \cdots & 0 & \\ 1 & 1 & 0 & 0 & \cdots & 0 & \\ 0 & 1 & 1 & 0 & \cdots & 0 & \\ 0 & 0 & 1 & 1 & \cdots & 0 & \text{other columns} \\ & & & & \ddots & & \\ 0 & 0 & 0 & 0 & \cdots & 1 & \\ 0 & 0 & 0 & 0 & \cdots & 1 & \end{array}\right]. \tag{17.87}$$

Masking the base matrix $\mathbb{H}_{EG,GA}(\rho)$ given by (17.72) with $\mathbb{Z}$, we obtain the masked matrix

$$\mathbb{M} = \mathbb{Z} \otimes \mathbb{H}_{EG,GA}(\rho)$$

with column and row weight distributions $\mathbb{c}(X)$ and $\mathbb{d}(X)$, respectively. Consequently, the associated Tanner graph of $\mathbb{M}$ has variable- and check-node degree distributions

$\gamma(X)$ and $\rho(X)$, as designed. Hence, the null space of $\mathbb{M}$ gives an irregular LDPC code with the designed degree distributions. Because the size of $\mathbb{Z}$ is, in general, very small compared with the size of the code graph, it is much easier to construct to meet the degree distribution requirements. Furthermore, since the associated Tanner graph of the base matrix $\mathbb{H}_{EG,GA}(\rho)$ is already free of cycles of length 4, we do not have to worry about eliminating these short cycles. The masking construction significantly reduces the computation and search effort of large random edge selection (or assignment) in the construction of the code graph, as is needed in computer generation of an irregular LDPC code [12].

We now use two examples to illustrate the construction of irregular LDPC codes using the masking technique.

EXAMPLE 17.32

Suppose an irregular LDPC code with desired length $n_d = 4000$ and rate 0.82 is to be constructed. Using the density evolution technique, we find the asymptotically optimal variable- and check-node degree distributions for rate 0.82:

$$\gamma(X) = 0.4052X + 0.3927X^2 + 0.1466X^6 + 0.0555X^7,$$
$$\rho(X) = 0.3109X^{18} + 0.6891X^{19}.$$

The maximum variable- and check-node degrees are 8 and 20, respectively. Suppose we choose the two-dimensional Euclidean geometry EG$(2, 2^6)$ over $GF(2^6)$ for constructing the base matrix $\mathbb{H}_{EG,GA}(\rho)$ given by (17.72). Using this geometry, we can partition the incidence matrix of a parallel bundle of lines into sixty-four 64×64 permutation matrices. To achieve a code length close to the desired code length 4000, we choose $\rho = 63$, which is larger than the maximum check-node degree 20. To achieve a rate close to the desired code rate 0.82, we choose $\gamma = 12$, which is greater than the maximum variable-node degree 8. For the choice of $\gamma = 12$ and $\rho = 63$, the rate of the code is at least $(\rho - \gamma)/\rho = (63 - 12)/64 = 0.81$, which is close to the desired rate 0.82. The length of the code is $63 \times 64 = 4032$, which is close to the desired code length. It follows from (17.72) that the base matrix $\mathbb{H}_{EG,GA}(63)$ is a 768×4032 matrix, which is a 12×63 array of 64×64 permutation matrices. For the desired code, the number of parity-check bits is $4000 \times (1 - 0.82) = 720$; however, the number of degree-2 variable nodes computed using the first coefficient $\gamma_2 = 0.4052$ of $\gamma(X)$ is $n_d \times \gamma_2 = 4000 \times 0.4052 = 1620$, which is larger than 720, the number of parity-check bits of the desired code. Therefore, we modify the variable-node degree distribution $\gamma(X)$ by converting 903 variable nodes of degree 2 to degree 3. The degree redistribution results in a new variable-node degree distribution,

$$\gamma^*(X) = 0.1798X + 0.6181X^2 + 0.1466X^6 + 0.0555X^7.$$

There is a small change in the coefficient of each degree in the check-node degree distribution $\rho(X)$. Next, we need to construct a 12×63 masking matrix $\mathbb{Z}$ with column and row weight distributions identical to the modified degree distributions $\gamma^*(X)$ and $\rho^*(X)$. Table 17.5 gives the desired column and row weight distributions of the masking matrix $\mathbb{Z}$. By computer search, we find the desired masking matrix $\mathbb{Z}$.

Masking the 768×4032 base matrix $\mathbb{H}_{EG,GA}(63)$ with $\mathbb{Z}$, we obtain the masked matrix $\mathbb{M} = \mathbb{Z} \otimes \mathbb{H}_{EG,GA}(63)$. The null space of $\mathbb{M}$ gives a (4032, 3264) irregular code

TABLE 17.5: Desired column and row weight distributions of the masking matrix $\mathbb{Z}$ for Example 17.32.

Column weight distribution		Row weight distribution	
Column weight	**No. of columns**	**Row weight**	**No. of rows**
2	11	19	4
3	39	20	8
7	9		
8	4		

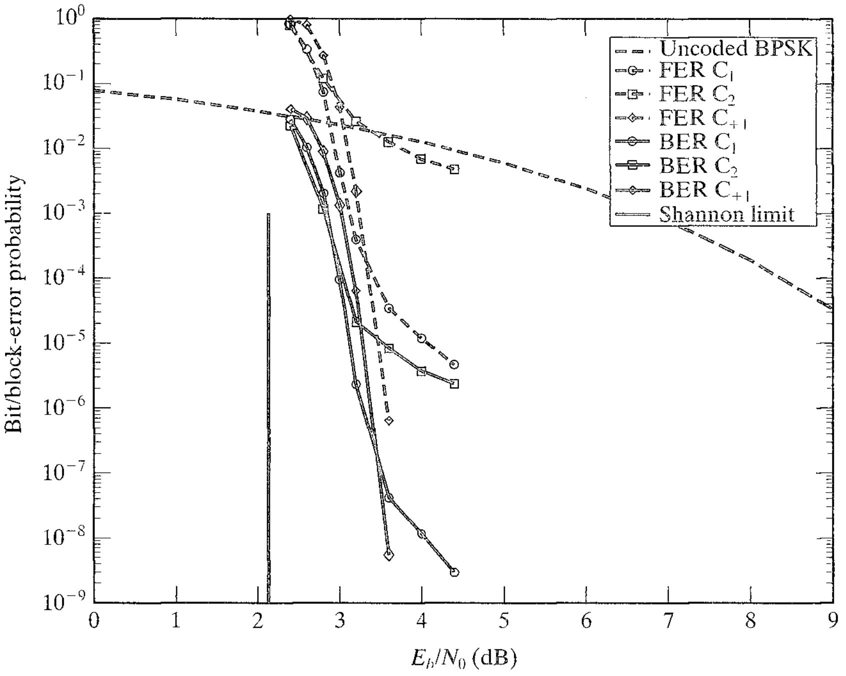

FIGURE 17.38: Error performances of the three (4032, 3264) irregular LDPC codes given in Examples 17.32 and 17.33.

C_1 with rate 0.81. The error performance of this code with SPA decoding is shown in Figure 17.38. At a BER of 10^{-5}, the code performs only 1 dB from the Shannon limit; however, it has an error floor starting around a BER of 10^{-6}. Figure 17.38 also includes the error performance of a (4032, 3264) irregular code C_2 constructed based on the original asymptotically optimal degree distributions without degree redistribution. We see that C_1 performs much better than C_2. The error floor of C_2 occurs above a BER of 10^{-5}.

An irregular code of finite length whose Tanner graph contains degree-2 variable nodes, in general, has an error floor owing to its relatively poor minimum distance. The error floor can be pushed down by either reducing the number of degree-2 variable nodes or removing all of them. An approach to removing all the degree-2 variable nodes is to increase every degree in the degree distribution $\gamma(X)$ of variable nodes by 1; that is, degree-2 is converted into degree-3, degree-3 is converted into degree-4, and so on [53].

EXAMPLE 17.33

(Example 17.32 continued) Suppose we increase every degree in the asymptotically optimal variable-node degree distribution $\gamma(X)$ given in Example 17.32 by 1. This degree conversion results in the following variable-node degree distribution:

$$\gamma_{+1}(X) = 0.4052X^2 + 0.3927X^3 + 0.1466X^7 + 0.0555X^8.$$

We must also modify the check-node degree distribution $\rho(X)$ to make the sum of check-node degrees the same as the sum of variable-node degrees in the new variable-node degree distribution $\gamma_{+1}(X)$. We can do this by increasing each degree in $\rho(X)$ by 4 without changing its coefficient. The result is a new check-node degree distribution:

$$\rho_{+1}(X) = 0.3109X^{22} + 0.6891X^{23}.$$

Using the new degree distributions, $\gamma_{+1}(X)$ and $\rho_{+1}(X)$, we construct a 12×63 masking matrix $\mathbb{Z}_{+1}$ with desired column and row weight distributions given in Table 17.6. For masking, the base matrix $\mathbb{H}_{EG,GA}(63)$ is still the same as the one given in Example 17.32. The masked matrix $\mathbb{M}_{+1} = \mathbb{Z}_{+1} \circledast \mathbb{H}_{EG,GA}(63)$ also gives a (4032, 3264) irregular LDPC code C_{+1}. The error performance of this code is also shown in Figure 17.38. We see that C_{+1} has much better error floor performance than codes C_1 and C_2, given in Example 17.32. Actually, there is no error floor down to a BER of 5×10^{-9}; however, there is a small performance degradation (less than 0.15 dB) above a BER of 10^{-7}.

Examples 17.32 and 17.33 show that pushing the error floor down by increasing the degrees of variable nodes of the code graph, the error performance of the code moves away from the Shannon limit in the waterfall region.

TABLE 17.6: Column and row weight distributions of the masking matrix for Example 17.33.

Column weight distribution		Row weight distribution	
Column weight	No. of columns	Row weight	No. of rows
3	25	23	4
4	25	24	8
8	9		
9	4		

EXAMPLE 17.34

The following optimal degree distributions of variable and check nodes of a Tanner graph are designed for a rate-1/2 irregular LDPC code of infinite length [12]:

$$\begin{aligned}\gamma(X) &= 0.47707681X + 0.28057X^2 + 0.034996X^3 + 0.096329X^4 \\ &\quad + 0.009088X^6 + 0.0013744X^{13} + 0.100557X^{14}, \\ \rho(X) &= 0.982081X^7 + 0.0179186X^8.\end{aligned}$$

Suppose we want to construct a rate-1/2 irregular LDPC code with length around 10000 using masking based on these degree distributions. To construct such a code, there are many geometries that can be used for constructing the base matrix for masking. We choose the two-dimensional Euclidean geometry EG$(2, 2^7)$ for constructing the base matrix $\mathbb{H}_{EG,GA}(\rho)$ given by (17.72). Using this geometry, we can partition the incidence matrix of a parallel bundle of lines into one hundred twenty-eight 128×128 permutation matrices. Suppose we choose $\gamma = 40$ and $\rho = 80$, which are greater than the maximum variable- and check-node degrees, 15 and 9, respectively. With the choice of $\gamma = 40$ and $\rho = 80$, $80 \times 128 = 10240$, which is close to the desired code length 10000, and $(\rho - \gamma)/\gamma = (80 - 40)/80 = 0.5$, which is the same as the desired rate 1/2. It follows from (17.72) that the base matrix $\mathbb{H}_{EG,GA}(80)$ is a 40×80 array of 128×128 permutation matrices. For masking, we need to construct a 40×80 masking matrix $\mathbb{Z}$ with desired column and row weight distributions given in Table 17.7. This is done by computer search. Masking $\mathbb{H}_{EG,GA}(80)$ with $\mathbb{Z}$, we obtain a 5120×10240 masked matrix $\mathbb{M} = \mathbb{Z} \otimes \mathbb{H}_{EG,GA}(80)$. The null space of $\mathbb{M}$ gives a (10240, 5102) irregular LDPC code C with rate 1/2. The error performance of this code is shown in Figure 17.39. At a BER of 10^{-6}, it performs 1 dB from the Shannon limit.

TABLE 17.7: Column and row weight distributions of the masking matrix for Example 17.34.

Column weight distribution		**Row weight distribution**	
Column weight	**No. of columns**	**Row weight**	**No. of rows**
2	38	8	39
3	22	9	1
4	3		
5	8		
7	1		
15	8		

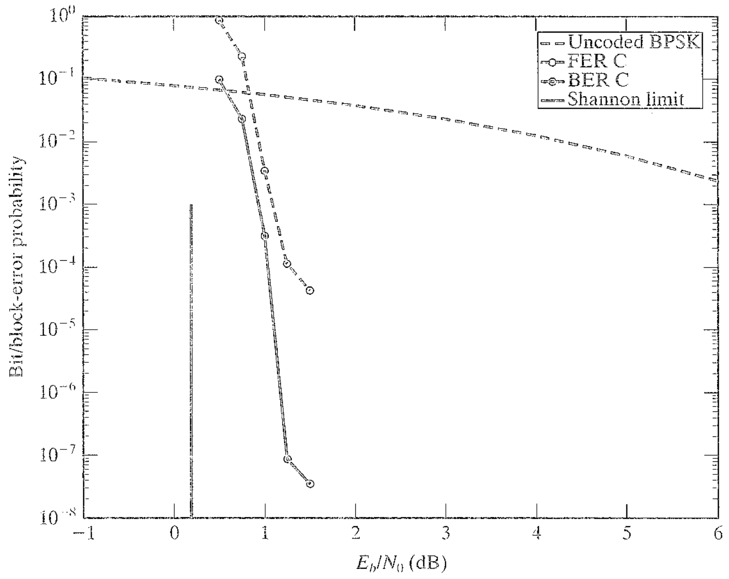

FIGURE 17.39: Error performances of the two (10240, 5120) irregular LDPC codes given in Example 17.34.

17.16 GRAPH-THEORETIC LDPC CODES

Finite geometries form a branch of combinatorial mathematics. Besides finite geometries, there are other branches in combinatorial mathematics that can be used for constructing LDPC codes. These branches include *graph theory*, *combinatoric designs*, and *difference sets* [35, 41, 57, 58]. In Section 8.3 we showed that codes constructed based on perfect difference sets are one-step MLG decodable, and we also showed in Section 17.5 that the codes constructed based on a special class of perfect difference sets are LDPC codes whose Tanner graphs do not contain cycles of length 4. Construction of LDPC codes based on random bipartite (or Tanner) graphs was briefly discussed in the previous section of this chapter. Construction based on random graphs results in an ensemble of LDPC codes. In this section we present another graph-theoretic approach to the construction of LDPC codes [59]. The construction is based on selecting a set of paths of the same length in a given graph that satisfies certain constraints.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a connected graph with vertex set $\mathcal{V} = \{v_1, v_2, \cdots, v_q\}$ and edge set $\mathcal{E} = \{e_1, e_2, \cdots, e_L\}$. We require that $\mathcal{G}$ not contain self-loops, and two vertices in $\mathcal{G}$ are connected by at most one edge (i.e., no multiple edges between two vertices). Some basic structural properties of a graph were discussed in Section 17.2. Two paths in $\mathcal{G}$ are said to be *disjoint* if they do not have any vertex in common. Two paths are said to be *singularly crossing* each other if they have one and only one vertex in common; that is, they intersect (or cross each other) at one and only one vertex. For $1 < \gamma \leq q$, let $\mathcal{P}$ be a set of paths of length $\gamma - 1$ in $\mathcal{G}$, that satisfies the following constraint:

Any two paths in $\mathcal{P}$ are either disjoint or singularly crossing each other.

This constraint is called the *disjoint-crossing* (DC) constraint. Let $n = |\mathcal{P}|$ denote the number of paths in $\mathcal{P}$ and q_o denote be the number of vertices in $\mathcal{G}$ that are covered by (or on) the paths in $\mathcal{P}$. For simplicity, we call the vertices in $\mathcal{G}$ that are covered by the paths in $\mathcal{P}$ the vertices in (or of) $\mathcal{P}$. We form a $q_o \times n$ matrix $\mathbb{H} = [h_{i,j}]$ whose rows correspond to the q_o vertices in $\mathcal{P}$ and whose columns correspond to the n paths in $\mathcal{P}$, where $h_{i,j} = 1$ if and only if the ith vertex v_i of $\mathcal{P}$ is on the jth path in $\mathcal{P}$; otherwise, $h_{i,j} = 0$. This matrix is called the *incidence matrix* of $\mathcal{P}$. The columns of this matrix are called the *incidence vectors* of the paths in $\mathcal{P}$ with respect to the vertices in $\mathcal{P}$; the jth column simply displays the vertices on the jth path of $\mathcal{P}$. Because the length of each path in $\mathcal{P}$ is $\gamma - 1$, there are γ vertices on each path in $\mathcal{P}$. Therefore, each column of $\mathbb{H}$ has weight γ. The weight of the ith row of $\mathbb{H}$ is equal to the number of paths in $\mathcal{P}$ that intersect at the ith vertex v_i of $\mathcal{P}$. It follows from the DC constraint that no two columns (or two rows) of $\mathbb{H}$ can have more than one 1-component in common. If γ is much smaller than the number q_o of vertices in $\mathcal{P}$, $\mathbb{H}$ is a sparse matrix. Then, the null space C over $GF(2)$ of $\mathbb{H}$ gives an LDPC code of length n whose Tanner graph does not contain cycles of length 4. The minimum distance of C is at least $\gamma + 1$. If the rows of $\mathbb{H}$ have the same weight ρ (i,e., each vertex of $\mathcal{P}$ is intersected by ρ paths in $\mathcal{P}$), then C is a (γ, ρ)-regular LDPC code.

The path set $\mathcal{P}$ can be constructed by using a trellis $\mathcal{T}$ of $\gamma - 1$ sections with γ levels of nodes, labeled $0, 1, \cdots, \gamma - 1$. Each level of $\mathcal{T}$ consists of q nodes that are simply the q vertices of $\mathcal{G}$. For $0 \leq k < \gamma - 1$, a node v_i at the kth level of $\mathcal{T}$ and a node v_j at the $(k+1)$th level of $\mathcal{T}$ are connected by a branch if and only if (v_i, v_j) is an edge in $\mathcal{G}$. This $(\gamma - 1)$-section trellis $\mathcal{T}$ is called the *path trellis* of $\mathcal{G}$ of length $\gamma - 1$. A path of length $\gamma - 1$ in $\mathcal{G}$ is a path in $\mathcal{T}$ starting from an initial node at the 0th level of $\mathcal{T}$ and ending at a terminal node at the $(\gamma - 1)$th level of $\mathcal{T}$ such that all the nodes on the path are different. Figures 17.40(a) and 17.40(b) show a complete graph (i.e., every two vertices are connected by an edge) with seven vertices and its path trellis of length 2, respectively.

To find a set $\mathcal{P}$ of paths of length $\gamma - 1$ in $\mathcal{G}$ that satisfies the DC constraint, an *extend-select-eliminate* (ESE) algorithm [59] can be devised to parse the path trellis $\mathcal{T}$ of $\mathcal{G}$. Suppose we have parsed the trellis $\mathcal{T}$ up to the kth level of $\mathcal{T}$ with the ESE algorithm, with $0 \leq k < \gamma - 1$. For $1 \leq i \leq q$, let $N_{i,k}$ denote the number of survivor paths of length k terminating at the ith node of the kth level of $\mathcal{T}$ (i.e., the paths that satisfies the DC constraint). Let $P_{i,k}$ denote this set of survivor paths. Now, we extend all the paths in $P_{i,k}$ to the $(k+1)$th level of $\mathcal{T}$ through the branches diverging from node i. At node i of the $(k+1)$th level of $\mathcal{T}$, we select a set $P_{i,k+1}$ of paths of length $k+1$ that satisfies the DC constraint and eliminate all the other paths that terminate at node i. The result is a survivor path set $P_{i,k+1}$ at node i of level $k+1$ of $\mathcal{T}$. This ESE process continues until the $(\gamma - 1)$th level of $\mathcal{T}$ is reached. Then, the union of the survivor sets, $P_{1,\gamma-1}, P_{2,\gamma-1}, \cdots, P_{q,\gamma-1}$, gives the path set $\mathcal{P}$. For $1 \leq i \leq q$ and $0 \leq k < \gamma$, all the survivor paths in $P_{i,k}$ start from different initial nodes at the 0th level of $\mathcal{T}$ and intersect only at node i of the kth level of $\mathcal{T}$. Two survivor paths terminating at two different nodes at the kth level of $\mathcal{T}$ are either disjoint or cross each other only once at a node of an earlier level of $\mathcal{T}$. Selection of

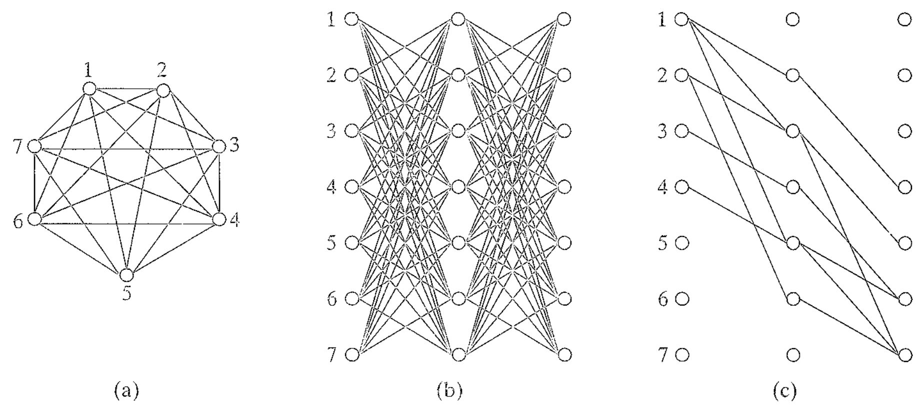

FIGURE 17.40: Complete graph of seven vertices, its path trellis of length 2, and a set of paths of length 2 satisfying the DC constraint found by the ESE algorithm.

survivor paths terminating at one node of the kth level of $\mathcal{T}$ affects the selection of survivor paths terminating at the other nodes of the same level of $\mathcal{T}$. To maintain all the sets of survivor paths at each level of $\mathcal{T}$ at about the same size, we create a priority list of the nodes at each level. The node with the smallest number of survivor paths terminating at it at level k of $\mathcal{T}$ has the first priority of survivor path selection at level $k+1$ of $\mathcal{T}$, whereas the one with the largest number of survivor paths terminating at it at level k of $\mathcal{T}$ has the lowest priority for the survivor path selection at level $k+1$ of $\mathcal{T}$.

EXAMPLE 17.35

Applying the ESE algorithm to the 2-section path trellis of the complete graph $\mathcal{G}$ of seven vertices shown in Figure 17.40 (b), we show a set $\mathcal{P}$ of paths of length 2 that satisfies the DC constraint in Figure 17.40(c). Expressing each path in terms of the vertices lying on it, we see that the paths are

$$(1, 2, 4), (1, 3, 7), (1, 5, 6), (2, 3, 5), (2, 6, 7), (3, 4, 6), (4, 5, 7).$$

We see that the paths in $\mathcal{P}$ cover all seven vertices of the graph. The incidence matrix for this set of paths is

$$\mathbb{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}, \tag{17.88}$$

which has constant row and column weights. The null space of $\mathbb{H}$ gives a (3, 3)-regular (7, 3) LDPC code with a minimum distance of 4 whose Tanner graph does not contain cycles of length 4, but it does contain cycles of length 6. One such cycle is depicted by the heavy lines shown in Figure 17.41.

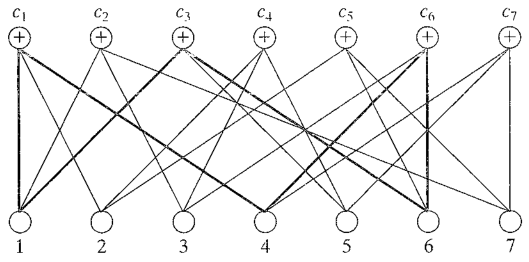

FIGURE 17.41: The Tanner graph of the (3, 3)-regular LDPC code generated by the parity-check matrix given by (17.88).

The graph-theoretic construction of LDPC codes given here is very general and can be applied to any connected graph $\mathcal{G}$. Because the selection of survivor paths at a node of each level of the path trellis $\mathcal{T}$ of $\mathcal{G}$ is not unique, the path set $\mathcal{P}$ obtained at the end of the ESE processing is not unique, and hence the code constructed is not unique.

EXAMPLE 17.36

Suppose we start with a complete graph $\mathcal{G}$ with 400 vertices. We choose $\gamma = 6$. The path trellis $\mathcal{T}$ of $\mathcal{G}$ is a 5-section trellis with six levels of nodes, with each level of $\mathcal{T}$ consisting of 400 nodes. Each path of length 5 in $\mathcal{G}$ is a path in $\mathcal{T}$. Applying the ESE algorithm to process $\mathcal{T}$, we obtain a set $\mathcal{P}$ of 2738 paths of length 5 that satisfies the DC constraint. The paths in $\mathcal{P}$ cover all 400 vertices of the graph. The incidence matrix $\mathbb{H}$ of $\mathcal{P}$ is a 400×2738 matrix with column weight 6. The null space of $\mathbb{H}$ gives a (2738, 2339) LDPC code with rate 0.8543 and a minimum distance of at least 7. The bit- and block-error performances of this code with SPA decoding are shown in Figure 17.42. At a BER of 10^{-6}, the code performs only 1.7 dB from the Shannon limit. Note that the number of parity bits of the code is one less than the number of vertices of the code construction graph.

For a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a large number of vertices, especially a complete graph, processing the path trellis of $\mathcal{G}$ of length $\gamma - 1$ to construct a set of paths of length $\gamma - 1$ in $\mathcal{G}$ to satisfy the DC constraint with the ESE algorithm may become very complex. To overcome this complexity problem, we can take a divide-and-conquer approach [59]. Consider a complete graph $\mathcal{G}$ with q vertices. Suppose we want to construct a set $\mathcal{P}$ of paths of length $\gamma - 1$ in $\mathcal{G}$ that satisfies the DC constraint. We first divide $\mathcal{G}$ into γ blocks, $\mathcal{G}_1, \mathcal{G}_1, \cdots, \mathcal{G}_\gamma$, of equal (or nearly equal) number of vertices. Each block $\mathcal{G}_i$ is a complete subgraph of $\mathcal{G}$, and any two blocks are connected by edges. For each block $\mathcal{G}_i$ we construct a set $\mathcal{P}_i$ of paths of length $\gamma - 1$ in $\mathcal{G}_i$ that satisfies the DC constraint using the ESE algorithm. Clearly, $\mathcal{P}_1, \mathcal{P}_1, \cdots, \mathcal{P}_\gamma$ are disjoint, and two paths from two different sets, $\mathcal{P}_i$ and $\mathcal{P}_j$, do not have any vertex in common. Next, we form a trellis $\mathcal{T}^c$ with γ levels of vertices, labeled $0, 1, \cdots, \gamma - 1$. For $0 \leq i < \gamma$, the ith level of $\mathcal{T}^c$ consists of q_{i+1} nodes that correspond to the q_{i+1} vertices in block $\mathcal{G}_{i+1}$. For $0 \leq i < \gamma - 1$, the nodes at the ith level of $\mathcal{T}^c$ are connected to the nodes at the $(i + 1)$th level of $\mathcal{T}^c$ by

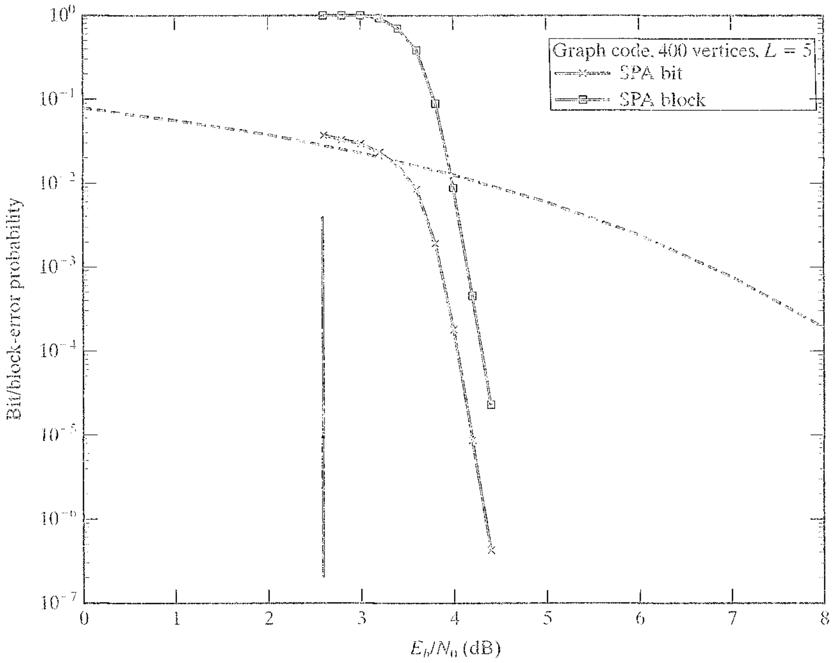

FIGURE 17.42: Error performance of the (2738, 2339) LDPC code constructed based on the complete graph with 400 vertices using the ESE algorithm.

branches that are the edges connecting the vertices in $\mathcal{G}_{i+1}$ to the vertices in $\mathcal{G}_{i+2}$. This $(\gamma - 1)$-section trellis is called a *connecting trellis* for the blocks of $\mathcal{G}$. Then, we process $\mathcal{T}^c$ with the ESE algorithm to find a set $\mathcal{P}^c$ of paths of length $\gamma - 1$ in $\mathcal{T}^c$ that satisfies the DC constraint. Every path in $\mathcal{P}^c$ is a path in $\mathcal{G}$ that contains one and only one vertex in each of the γ blocks of $\mathcal{G}$, and the ith vertex is in $\mathcal{G}_i$. The paths in $\mathcal{P}^c$ are called *connecting paths* of the blocks of $\mathcal{G}$. A path in $\mathcal{P}_i$ and a connecting path in $\mathcal{P}^c$ are either disjoint or they singularly cross each other. For each connecting path in $\mathcal{P}^c$, there is at least one path in $\mathcal{P}_i$ that singularly crosses with it. The union

$$\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2 \cup \cdots \cup \mathcal{P}_\gamma \cup \mathcal{P}^c$$

gives a set of paths in $\mathcal{G}$ that satisfies the DC constraint. The complexity of processing the connecting trellis $\mathcal{T}^c$ with the ESE algorithm is about the same as that of processing the path trellis of length $\gamma - 1$ of each block with the ESE algorithm. We form the incidence matrix $\mathbb{H}$ for $\mathcal{P}$. Then, the null space of $\mathbb{H}$ gives an LDPC code whose Tanner graph does not contain cycles of length 4.

EXAMPLE 17.37

Consider the complete graph $\mathcal{G}$ with 582 vertices. Let $\gamma = 6$. We divide $\mathcal{G}$ into six blocks; each block is a complete graph with 97 vertices. Using the divide-and-conquer ESE algorithm to process the path trellis of length 5 for each block and the connecting trellis for the blocks, we obtain a set $\mathcal{P}$ of 10429 paths of length 5 that satisfies the DC constraint and covers all 582 vertices of the graph. We form the incidence matrix $\mathbb{H}$ for $\mathcal{P}$, which is a 582×10429 matrix. The null space of $\mathbb{H}$ gives a (10429, 9852) LDPC code with rate 0.945 and a minimum distance of at least 7 whose Tanner graph does not contain cycles of length 4. The bit- and block-error performances with SPA decoding are shown in Figure 17.43. At a BER of 10^{-6}, the code performs 1.05 dB from the Shannon limit.

We also can construct a path set $\mathcal{P}$ of length $\gamma - 1$ in a connected graph $\mathcal{G}$ that satisfies the DC constraint by choosing one path at a time through the $(\gamma - 1)$-section path trellis $\mathcal{T}$ of $\mathcal{G}$. We begin by choosing any path in $\mathcal{T}$ as the first path in $\mathcal{P}$. We delete all the branches on this path from $\mathcal{T}$. Then, we trace the modified path trellis and choose the second path that either has no common vertex with the first path in $\mathcal{P}$ or intersects with the first path in $\mathcal{P}$ at one and only one vertex (DC constraint). Again, we delete the branches on the second path from the modified path trellis and obtain a new modified path trellis. Then, we trace this new path trellis and choose the third path that satisfies the DC constraint with the two paths in $\mathcal{P}$. We delete the branches on the third path from the path trellis and then start a new path selection. We continue this process of path selection and removal of branches on a selected path from the path trellis. Each time we select a path it must satisfy the DC constraint with the paths already in $\mathcal{P}$. When a node in the path trellis has no incoming or outgoing branches, it is removed from the path trellis together with all the branches still attached to it. We continue this process until no path in the path trellis can be chosen without violating the DC constraint, or the path trellis

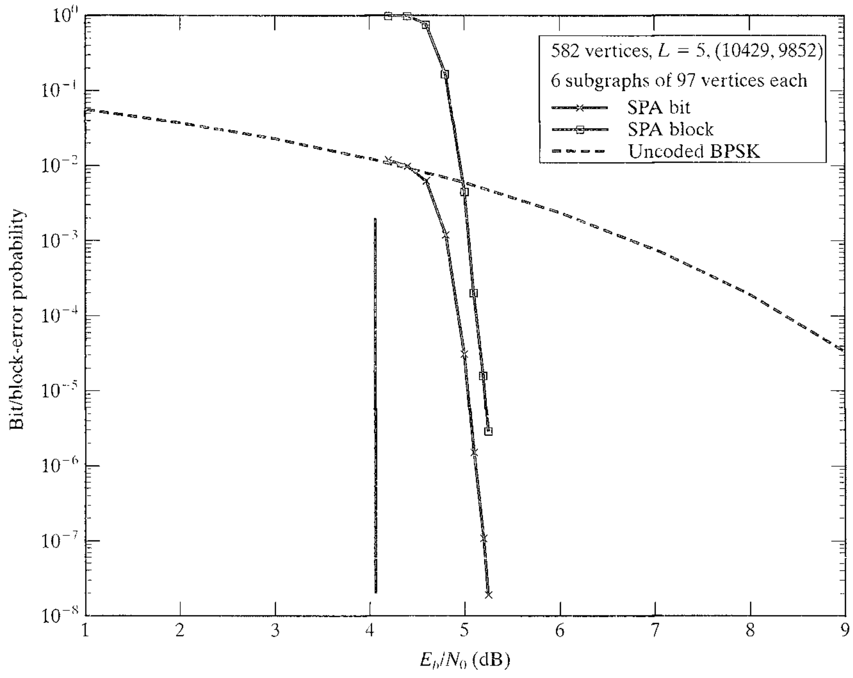

FIGURE 17.43: Error performance of the (10429, 9852) LDPC code constructed based on the complete graph of 582 vertices using the divide-and-conquer ESE algorithm.

is completely disconnected. The resultant path set $\mathcal{P}$ gives an LDPC code. This code construction procedure is actually a graph-theoretic model of the first random LDPC code construction procedure presented in Section 17.14 with the complete graph of $n - k$ vertices as the code construction graph.

Other constructions of LDPC codes based on specific classes of graphs with large girths have also been proposed [60–62].

17.17 CONSTRUCTION OF LDPC CODES BASED ON BALANCED INCOMPLETE BLOCK DESIGNS

Combinatoric design is another important subject in combinatorial mathematics. The objective of this subject is to design experiments systematically and with a view to their analysis. One such design is known as *balanced incomplete block design* (BIBD) [41, 58]. A special subclass of BIBDs can be used for constructing of LDPC codes.

Let $X = \{x_1, x_2, \cdots, x_q\}$ be a set of q objects. A BIBD of X is a collection of n γ-subsets of X, denoted by $B_1, B_2, \cdots, B_n$, called *blocks*, such that the following conditions are satisfied: (1) each object appears in exactly ρ of the n blocks; (2) every two objects appear together in exactly λ of the n blocks; and (3) the number γ of objects in each block is small compared with the total number of objects in X. Thus, a BIBD is characterized by five parameters, n, q, γ, ρ, and λ. Instead of being described by a list of the blocks, a BIBD can be described by a $q \times n$ matrix $\mathbb{H} = [h_{i,j}]$ whose rows correspond to the q objects of X and whose columns correspond to the n blocks of the design, where $h_{i,j} = 1$ if and only if the ith object x_i is in the jth block, and $h_{i,j} = 0$ otherwise. This matrix $\mathbb{H}$ is called the incidence matrix of the design. The column and the row weights of $\mathbb{H}$ are γ and ρ, respectively. Based on the second condition of a BIBD, two rows of $\mathbb{H}$ have exactly λ 1-components in common. If $\lambda = 1$, then $\mathbb{H}$ meets all the conditions of a regular parity-check matrix of an LDPC code given by Definition 17.1. Then, the null space of $\mathbb{H}$ gives a (γ, ρ)-regular LDPC code of length n whose Tanner graph is free of cycles of length 4.

EXAMPLE 17.38

Let $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$ be a set of seven objects. The following blocks:

$$\{x_1, x_2, x_4\}, \quad \{x_2, x_3, x_5\}, \quad \{x_3, x_4, x_6\}, \quad \{x_4, x_5, x_7\},$$
$$\{x_1, x_5, x_6\}, \quad \{x_2, x_6, x_7\}, \quad \{x_1, x_3, x_7\},$$

form a BIBD for the set X. Every block consists of $\gamma = 3$ objects, each object appears in $\rho = 3$ blocks, and every two objects appear together in exactly $\lambda = 1$ block. Its incidence matrix $\mathbb{H}$ is

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

The null space of $\mathbb{H}$ gives a (7, 3) LDPC code with a minimum distance of 4.

Combinatoric design is a very old and rich subject in combinatorial mathematics. Over the years, many BIBDs have been constructed with various methods. Extensive coverage can be found in [41, 58]. There are many classes of BIBDs with $\lambda = 1$, which can be used for constructing LDPC codes whose Tanner graphs do not contain cycles of length 4 [64, 65]. Construction of these classes of designs requires much combinatorial mathematics and finite-group theory background and will not be covered here; however, we do present one special class of BIBDs with parameter $\lambda = 1$ constructed by Bose [63].

Let t be a positive integer such that $20t + 1 = p^m$, where p is a prime; that is, $20t + 1$ is a power of a prime. Suppose the field $GF(p^m)$ has a primitive element α that satisfies the condition $\alpha^{4t} - 1 = \alpha^c$, where c is a positive odd integer less than p^m. Then, there exists a BIBD for a set X of $q = 20t + 1$ objects with $n = t(20t + 1)$ blocks, each block consists of $\gamma = 5$ objects, each object appears in $\rho = 5t$ blocks, and every two objects appear together in exactly $\lambda = 1$ block. Let the elements of $GF(p^m), 0, \alpha^0 = 1, \alpha, \alpha^2, \cdots, \alpha^{p^m-2}$, represent the $20t + 1 = p^m$ objects of the set X. Then, the BIBD for X is completely specified by the following t base blocks:

$$B_i = \{\alpha^{2i}, \alpha^{2i+4t}, \alpha^{2i+8t}, \alpha^{2i+12t}, \alpha^{2i+16t}\}, \tag{17.89}$$

where $0 \le i < t$. All the $n = t(20t + 1)$ blocks of the BIBD are obtained by adding each element of $GF(p^m)$ in turn to each of the t base blocks. The incidence matrix $\mathbb{H}$ of this BIBD is a $(20t + 1) \times t(20t + 1)$ matrix with column and row weights 5 and $5t$, respectively. The density of $\mathbb{H}$ is $5/(20t + 1)$, which is very small for $t \ge 2$. Because $\lambda = 1$, it follows from Definition 17.1 that the null space of $\mathbb{H}$ gives an LDPC code of length $n = t(20t + 1)$ whose Tanner graph is free of cycles of length 4. In fact, $\mathbb{H}$ can be put in circulant form. For $0 \le i < t$, let $\mathbf{v}_i$ be the incidence vector of the base block B_i which is a p^m-tuple with 1's at locations $2i$, $2i + 4t$, $2i + 8t$, $2i + 12t$, and $2i + 16t$. Let $\mathbb{G}_i$ be a $(20t + 1) \times (20t + 1)$ square circulant matrix obtain by shifting $\mathbf{v}_i$ downward cyclically $20t + 1$ times (including the zero shift). All the columns (or rows) of $\mathbb{G}_i$ are different and are the incidence vectors of $20t + 1$ different blocks. Then, $\mathbb{H}$ can be put into the following circulant form:

$$\mathbb{H} = [\mathbb{G}_0 \; \mathbb{G}_1 \; \cdots \; \mathbb{G}_{t-1}]. \tag{17.90}$$

With $\mathbb{H}$ in circulant form, the null space of $\mathbb{H}$ gives a quasi-cyclic BIBD-LDPC code of length $n = t(20t+1)$. For $1 \le k \le t$, we can choose k circulants, $\mathbb{G}_0, \mathbb{G}_1, \cdots, \mathbb{G}_{k-1}$, to form the following matrix:

$$\mathbb{H}(k) = [\mathbb{G}_0 \; \mathbb{G}_1 \; \cdots \; \mathbb{G}_{k-1}] \tag{17.91}$$

with column and row weights 5 and $5k$, respectively. Then, the null space of $\mathbb{H}(k)$ gives a quasi-cyclic LDPC code of length $n = k(20t + 1)$.

Suppose we decompose each circulant $\mathbb{G}_i$ in $\mathbb{H}(k)$ into five $(20t + 1) \times (20t + 1)$ circulant permutation matrices by row decomposition as presented in Section 15.12,

$$\mathbb{D}_i = \begin{bmatrix} \mathbb{G}_{1,i} \\ \mathbb{G}_{2,i} \\ \mathbb{G}_{3,i} \\ \mathbb{G}_{4,i} \\ \mathbb{G}_{5,i} \end{bmatrix}. \tag{17.92}$$

Replacing each circulant $\mathbb{G}_i$ in $\mathbb{H}(k)$ with its row decomposition $\mathbb{D}_i$, we obtain the following $5 \times k$ array of $(20t+1) \times (2t+1)$ permutation matrices:

$$\mathbb{D} = \begin{bmatrix} \mathbb{G}_{1,0} & \mathbb{G}_{1,1} & \cdots & \mathbb{G}_{1,k-1} \\ \mathbb{G}_{2,0} & \mathbb{G}_{2,1} & \cdots & \mathbb{G}_{2,k-1} \\ \mathbb{G}_{3,0} & \mathbb{G}_{3,1} & \cdots & \mathbb{G}_{3,k-1} \\ \mathbb{G}_{4,0} & \mathbb{G}_{4,1} & \cdots & \mathbb{G}_{4,k-1} \\ \mathbb{G}_{5,0} & \mathbb{G}_{5,1} & \cdots & \mathbb{G}_{5,k-1} \end{bmatrix}. \tag{17.93}$$

The null space of $\mathbb{D}$ gives a quasi-cyclic LDPC code with a minimum distance of at least 6 and rate of at least $(k-5)/k$. If we remove one row of permutation matrices from $\mathbb{D}$, the resultant submatrix of $\mathbb{D}$ generates a quasi-cyclic code also with a minimum distance of at least 6 but rate of at least $(k-4)/k$. If we move two rows from $\mathbb{D}$, then the resultant submatrix generates a quasi-cyclic LDPC code with a minimum distance of at least 4 and rate of at least $(k-3)/k$.

EXAMPLE 17.39

Let $t = 21$. Then, $20t + 1 = 421$, which is a prime. The field $GF(421)$ does have a primitive element α that satisfies the condition $\alpha^{4t} - 1 = \alpha^c$, with c as a positive odd integer less than 421 (see Problem 17.26). Therefore, there is a BIBD for a set X of 421 objects. This BIBD consists of $n = 8841$ blocks, each block consists of $\gamma = 5$ objects, each object appears in exactly 105 blocks, and $\lambda = 1$. The 21 base blocks are

$$B_i = \{\alpha^{2i}, \alpha^{2i+84}, \alpha^{2i+168}, \alpha^{2i+252}, \alpha^{2i+336}\}$$

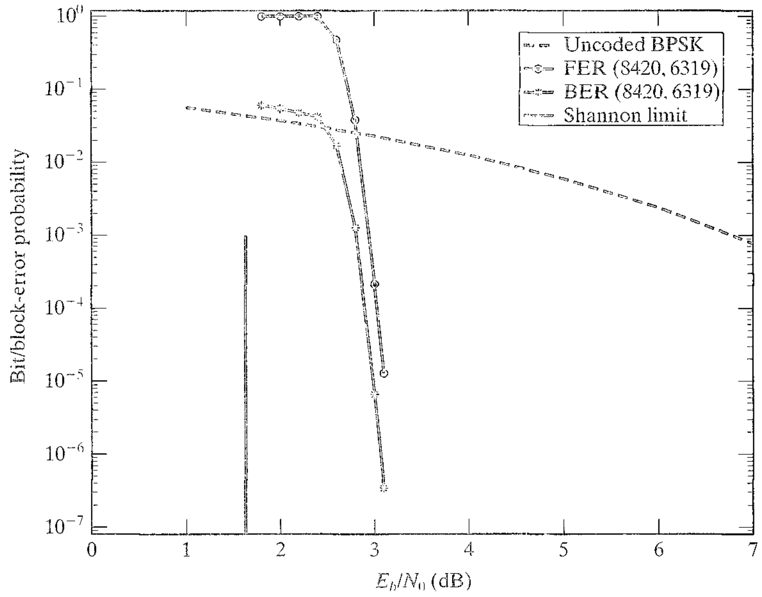

FIGURE 17.44: The error performance of the (8420, 6319) quasi-cyclic BIBD-LDPC code given in Example 17.39.

for $i = 0$ to 20. Based on these 21 base blocks, we can form the incidence matrix $\mathbb{H}$ for the BIBD in the following circulant forms:

$$\mathbb{H} = [\mathbb{G}_0, \mathbb{G}_1, \cdots, \mathbb{G}_{20}],$$

where $\mathbb{G}_i$ is a 421×421 circulant. Suppose we choose $k = 20$ and decompose each $\mathbb{G}_i$ into five circulant permutation matrices by row decomposition. We obtain a 5×20 array $\mathbb{D}$ of 421×421 circulant permutation matrices, which is a 2105×8420 matrix with column and row weights 5 and 20, respectively. The null space of $\mathbb{D}$ gives a (8420, 6319) quasi-cyclic BIBD-LDPC code with rate 0.7504 and a minimum distance of at least 6. The error performance of this code is shown in Figure 17.44. At a BER of 10^{-6}, it performs 1.4 dB from the Shannon limit.

17.18 CONSTRUCTION OF LDPC CODES BASED ON SHORTENED RS CODES WITH TWO INFORMATION SYMBOLS

In earlier sections of this chapter we used several branches of combinatorial mathematics as tools for constructing LDPC codes. In this section we present an algebraic method for constructing LDPC codes based on shortened RS codes with two information symbols [66]. This method gives a class of LDPC codes in Gallager's original form.

Let α be a primitive element of the Galois field $GF(q)$ where $q = p^s$ is a power of a prime. Let ρ be a positive integer such that $2 \leq \rho < q$. Then, the generator polynomial of the cyclic $(q-1, q-\rho+1, \rho-1)$ RS code C over $GF(q)$ of length $q-1$, dimension $q-\rho+1$, and minimum distance $\rho-1$ is given by (7.2),

$$\begin{aligned}\mathbf{g}(X) &= (X-\alpha)(X-\alpha^2)\cdots(X-\alpha^{\rho-2}) \\ &= g_0 + g_1X + g_2X^2 + \cdots + X^{\rho-2},\end{aligned}$$

where $g_i \in GF(q)$. The generator polynomial $\mathbf{g}(X)$ is a minimum-weight code polynomial in C, and hence all its $\rho-1$ coefficients are nonzero.

Suppose we shorten C by deleting the first $q-\rho-1$ information symbols from each codeword of C. Then, we obtain a $(\rho, 2, \rho-1)$ shortened RS code C_b with only two information symbols. A generator matrix of this shortened code is given by

$$\mathbb{G}_b = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & & 1 & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & & 1 \end{bmatrix}.$$

All the linear combinations of the two rows of $\mathbb{G}_b$ over $GF(q)$ give all the q^2 codewords of C_b. The nonzero codewords of C_b have two different weights, $\rho-1$ and ρ. Because the minimum distance of C_b is $\rho-1$, two codewords in C_b have at most one location with the same code symbol; that is, they agree at most at one location. Let $\mathbf{v}$ be a nonzero codeword in C_b with weight ρ. Then, the set

$$C_b^{(1)} = \{c\mathbf{v} : c \in GF(q)\}$$

of q codewords in C_b forms a one-dimensional subcode of C_b with minimum distance ρ. Two codewords in $C_b^{(1)}$ differ at every location. We partition C_b into q

cosets, $C_b^{(1)}, C_b^{(2)}, \cdots, C_b^{(q)}$, based on the subcode $C_b^{(1)}$. Then, two codewords in any coset $C_b^{(i)}$ must differ in all ρ locations, and two codewords in two different cosets, $C_b^{(i)}$ and $C_b^{(j)}$, differ at least at $\rho - 1$ locations. We arrange the q codewords of a coset $C_b^{(i)}$ as a $q \times \rho$ array $\mathbb{M}_i$. Then, all the q entries of any column of $\mathbb{M}_i$ are different (they are q different elements of $GF(q)$).

Let $\mathbb{z} = (z_\infty, z_0, z_1, \cdots, z_{q-2})$ be a q-tuple over $GF(2)$ whose components correspond to the q elements, $\alpha^\infty, \alpha^0, \alpha^1, \alpha^2, \cdots, \alpha^{q-2}$, of $GF(q)$; that is, z_i corresponds to the element α^i of $GF(q)$. We call α^i the *location number* of z_i. For $i = \infty, 0, 1, \cdots, q-2$, we define the *location vector* of the field element α^i as a q-tuple over $GF(2)$,

$$\mathbb{z}(\alpha^i) = (0, 0, \cdots, 0, 1, 0, 0, \cdots, 0), \tag{17.94}$$

for which the ith component of z_i is equal to 1, and all the other components are equal to zero. It follows from the definition that the 1-components of the location vectors of two different elements in $GF(q)$ are at two different locations. Suppose we arrange the location vectors of the q elements of $GF(q)$ as the rows of a matrix $\mathbb{A}$. Then, $\mathbb{A}$ is a $q \times q$ permutation matrix.

Let $\mathbb{v} = (v_1, v_2, \cdots, v_\rho)$ be a codeword in C_b. For $1 \le j \le \rho$, replacing each component v_j of $\mathbb{v}$ with its location vector $\mathbb{z}(v_j)$, we obtain a ρq-tuple over $GF(2)$,

$$\mathbb{z}(\mathbb{v}) = (\mathbb{z}(v_1), \mathbb{z}(v_2), \cdots, \mathbb{z}(v_\rho)). \tag{17.95}$$

Because the weight of each location vector $\mathbb{z}(v_i)$ is 1, the weight of $\mathbb{z}(\mathbb{v})$ is ρ. This ρq-tuple is called the *symbol location vector* of $\mathbb{v}$. Because any two codewords in C_b have at most one location with the same symbol, their symbol location vectors consequently have at most one 1-component in common. Let

$$\mathbb{Z}(C_b^{(i)}) = \{\mathbb{z}(\mathbb{v}) : \mathbb{v} \in C_b^{(i)}\} \tag{17.96}$$

be the set of symbol location vectors of the q codewords in the ith coset $C_b^{(i)}$ of $C_b^{(1)}$. Then, two symbol location vectors in $\mathbb{Z}(C_b^{(i)})$ do not have any 1-component in common, and two symbol location vectors from two different sets, $\mathbb{Z}(C_b^{(i)})$ and $\mathbb{Z}(C_b^{(j)})$, have at most one 1-component in common.

For $1 \le i \le q$, we form a $q \times \rho q$ matrix $\mathbb{H}_i$ over $GF(2)$ whose rows are the q symbol location vectors in $\mathbb{Z}(C_b^{(i)})$. Because the weight of each vector in $\mathbb{Z}(C_b^{(i)})$ is ρ, the total number of 1-entries in $\mathbb{H}_i$ is ρq. Since no two vectors in $\mathbb{Z}(C_b^{(i)})$ have any 1-component in common, the weight of each column of $\mathbb{H}_i$ is 1. Therefore, $\mathbb{H}_i$ has constant row weight ρ and constant column weight 1. It follows from the structural properties of the codewords in $C_b^{(i)}$ and their symbol location vectors that $\mathbb{H}_i$ consists of a row of ρ $q \times q$ permutation matrices,

$$\mathbb{H}_i = [\mathbb{A}_{i,1}\ \mathbb{A}_{i,2}\ \cdots\ \mathbb{A}_{i,\rho}]. \tag{17.97}$$

Matrix $\mathbb{H}_i$ is called the *symbol location matrix* of the coset $C_b^{(i)}$. The class of symbol location matrices,

$$\mathcal{H} = \{\mathbb{H}_1, \mathbb{H}_2, \cdots, \mathbb{H}_q\}, \tag{17.98}$$

has the following structural properties: (1) no two rows in the same matrix $\mathbb{H}_i$ have any 1-component in common; and (2) no two rows from two different matrices, $\mathbb{H}_i$ and $\mathbb{H}_j$, have more than one 1-component in common.

EXAMPLE 17.40

Consider the Galois field $GF(2^2)$ constructed based on the primitive polynomial $p(X) = 1 + X + X^2$. Let α be a primitive element of $GF(2^2)$. Then, the four elements of $GF(2^2)$ are $0 = \alpha^\infty$, $1 = \alpha^0$, α, and $\alpha^2 = 1 + \alpha$. The location vectors of these four field elements are

$$\mathbb{z}_0 = (1\,0\,0\,0), \qquad \mathbb{z}_1 = (0\,1\,0\,0), \qquad \mathbb{z}_\alpha = (0\,0\,1\,0), \qquad \mathbb{z}_{\alpha^2} = (0\,0\,0\,1).$$

Let $\rho = 3$. The cyclic (3, 2, 2) RS code C_b over $GF(2^2)$ has generator polynomial $\mathbb{g}(X) = X + \alpha$ and generator matrix

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

The code has 16 codewords, and its minimum distance is 2. Adding the two rows of $\mathbb{G}$, we obtain a codeword $\mathbb{v} = (\alpha, \alpha^2, 1)$ with weight 3. The following set of four codewords:

$$\{\beta\mathbb{v} : \beta \in GF(2^2)\} = \{(0, 0, 0), (\alpha, \alpha^2, 1), (\alpha^2, 1, \alpha), (1, \alpha, \alpha^2)\}$$

forms a one-dimensional subcode $C_b^{(1)}$ of C_b with a minimum weight of 3. We partition C_b with respect to $C_b^{(1)}$ and obtain the following four cosets:

$$\begin{aligned}
C_b^{(1)} &= \{(0, 0, 0), (\alpha, \alpha^2, 1), (\alpha^2, 1, \alpha), (1, \alpha, \alpha^2)\}, \\
C_b^{(2)} &= \{(\alpha, 1, 0), (0, \alpha, 1), (1, 0, \alpha), (\alpha^2, \alpha^2, \alpha^2)\}, \\
C_b^{(3)} &= (\alpha^2, \alpha, 0), (1, 1, 1), (0, \alpha^2, \alpha), (\alpha, 0, \alpha^2), \\
C_b^{(4)} &= (1, \alpha^2, 0), (\alpha^2, 0, 1), (\alpha, \alpha, \alpha), (0, 1, \alpha^2).
\end{aligned}$$

The symbol location matrices of these four cosets are

$$\mathbb{H}_1 = \mathbb{Z}(C_b^{(1)}) = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbb{H}_2 = \mathbb{Z}(C_b^{(2)}) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbb{H}_3 = \mathbb{Z}(C_b^{(3)}) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbb{H}_4 = \mathbb{Z}(C_b^{(4)}) = \left[\begin{array}{cccc cccc cccc} 0&1&0&0 & 0&0&0&1 & 1&0&0&0 \\ 0&0&0&1 & 1&0&0&0 & 0&1&0&0 \\ 0&0&1&0 & 0&0&1&0 & 0&0&1&0 \\ 1&0&0&0 & 0&1&0&0 & 0&0&0&1 \end{array}\right].$$

Let $\gamma = 3$. We form the following parity-check matrix:

$$\mathbb{H}_{GA}(3) = \left[\begin{array}{c} \mathbb{H}_1 \\ \mathbb{H}_2 \\ \mathbb{H}_3 \end{array}\right] = \left[\begin{array}{cccc cccc cccc} 1&0&0&0 & 1&0&0&0 & 1&0&0&0 \\ 0&0&1&0 & 0&0&0&1 & 0&1&0&0 \\ 0&0&0&1 & 0&1&0&0 & 0&0&1&0 \\ 0&1&0&0 & 0&0&1&0 & 0&0&0&1 \\ \hline 0&0&1&0 & 0&1&0&0 & 1&0&0&0 \\ 1&0&0&0 & 0&0&1&0 & 0&1&0&0 \\ 0&1&0&0 & 1&0&0&0 & 0&0&1&0 \\ 0&0&0&1 & 0&0&0&1 & 0&0&0&1 \\ \hline 0&0&0&1 & 0&0&1&0 & 1&0&0&0 \\ 0&1&0&0 & 0&1&0&0 & 0&1&0&0 \\ 1&0&0&0 & 0&0&0&1 & 0&0&1&0 \\ 0&0&1&0 & 1&0&0&0 & 0&0&0&1 \end{array}\right].$$

The null space $\mathbb{H}_{GA}(3)$ gives a regular (12, 4) RS-based Gallager-LDPC code with rate 1/3 and a minimum distance of 6. The lower bound $\gamma + 1$ on the minimum distance is 4.

Let γ be a positive integer such that $1 \le \gamma \le q$. We form the following $\gamma q \times \rho q$ matrix over $GF(2)$:

$$\mathbb{H}_{GA}(\gamma) = \left[\begin{array}{c} \mathbb{H}_1 \\ \mathbb{H}_2 \\ \vdots \\ \mathbb{H}_\gamma \end{array}\right] = \left[\begin{array}{cccc} \mathbb{A}_{1,1} & \mathbb{A}_{1,2} & \cdots & \mathbb{A}_{1,\rho} \\ \mathbb{A}_{2,1} & \mathbb{A}_{2,2} & \cdots & \mathbb{A}_{2,\rho} \\ & \cdots & & \cdots \\ \mathbb{A}_{\gamma,1} & \mathbb{A}_{\gamma,2} & \cdots & \mathbb{A}_{\gamma,\rho} \end{array}\right], \tag{17.99}$$

where each submatrix $\mathbb{A}_{i,j}$ is a $q \times q$ permutation matrix. Therefore, $\mathbb{H}_{GA}(\gamma)$ consists of a $\gamma \times \rho$ array of permutation matrices. It is a (γ, ρ)-regular matrix with column and row weights γ and ρ, respectively. No two rows (or two columns) of $\mathbb{H}_{GA}(\gamma)$ have more than one 1-component in common, and its density is $1/q$, which is small for large q. Hence, $\mathbb{H}_{GA}(\gamma)$ is a sparse matrix that has all the structural properties of the parity-check matrix of a regular LDPC code given in Definition 17.1. Furthermore, it is exactly in Gallager's original form given by (17.1). Therefore, the null space of $\mathbb{H}_{GA}(\gamma)$ gives an LDPC code $C_{GA}(\gamma)$ of Gallager's type of length $n = \rho q$ with a minimum distance of at least $\gamma + 1$ for odd γ and $\gamma + 2$ for even γ. The rate of this code is at least $(\rho - \gamma)/\rho$.

For any choice of q $(=p^s)$ and γ, we can construct a sequence of Gallager-LDPC codes of various lengths and rates with $\rho = \gamma, \gamma + 1, \cdots, q - 1$. For any choice of q and ρ, we can construct a sequence of Gallager-LDPC codes of length $n = \rho q$ with various rates and minimum distances for $\gamma = 1, 2, \cdots, \rho$. For $\rho = q - 1$, $C_{GA}(\gamma)$ is quasi-cyclic. Because the construction is based on the $(\rho, 2, \rho - 1)$ shortened RS code

C_b over the field $GF(q)$, we call C_b and $GF(q)$ the *base code* and the *construction field*, respectively.

EXAMPLE 17.41

Let $GF(2^6)$ be the construction field. Suppose we choose $\rho = 32$. Then, the base code is the (32, 2, 31) shortened RS code over $GF(2^6)$. The location vector of each element of $GF(2^6)$ is a 64-tuple with a single 1-component. Suppose we choose $\gamma = 6$. Then, the RS-based Gallager-LDPC code $C_{GA}(6)$ constructed is a (6, 32)-regular (2048, 1723) LDPC code with rate 0.841 and a minimum distance of at least 8. Its error performance with SPA decoding is shown in Figure 17.45. At a BER of 10^{-6}, the code performs 1.55 dB from the Shannon limit and achieves a 6-dB coding gain over the uncoded BPSK. If we choose $\rho = 63$, the base code is then the (63, 2, 62) shortened RS code. We set $\gamma = 60$. The RS-based Gallager-LDPC code is a (60, 63)-regular (4032, 3307) quasi-cyclic code with rate 0.82 and a minimum distance of at least 62. The error performance of this code with SPA decoding is shown in Figure 17.46. At a BER of 10^{-6}, it performs 1.65 dB from the Shannon limit. Owing to its large minimum distance, no error floor is expected. This code can also be decoded with one-step majority-logic decoding to correct 30 or fewer random errors. If it is decoded with weighted BF decoding, an effective trade-off between error performance and decoding complexity can be achieved.

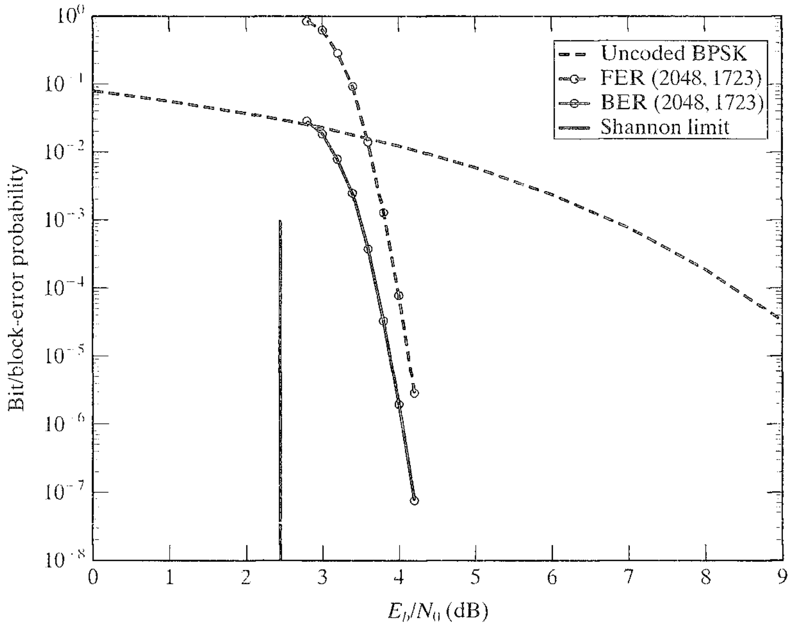

FIGURE 17.45: Error performance of the (2048, 1723) RS-based Gallager (6, 32)-regular LDPC code with construction field $GF(2^6)$ given in Example 17.41.

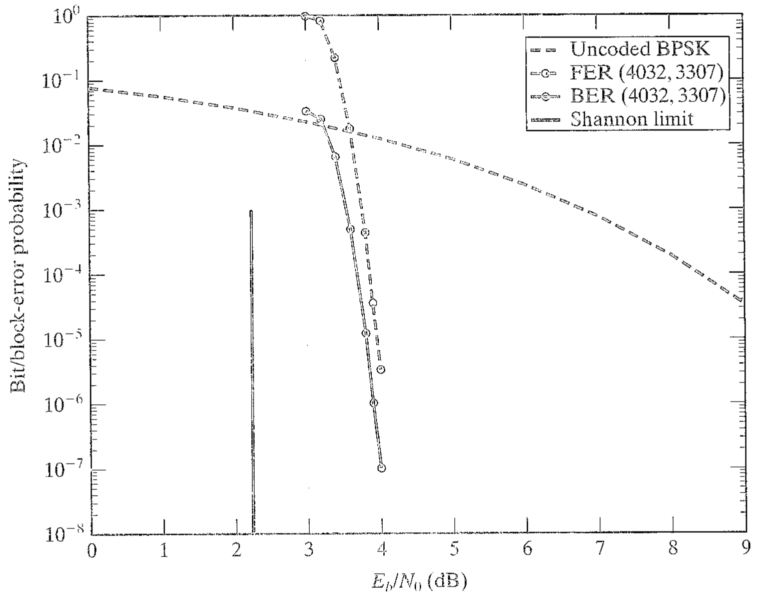

FIGURE 17.46: Error performance of the (4032, 3307) RS-based Gallager (60, 63)-regular quasi-cyclic LDPC code with construction field $GF(2^6)$ given in Example 17.41.

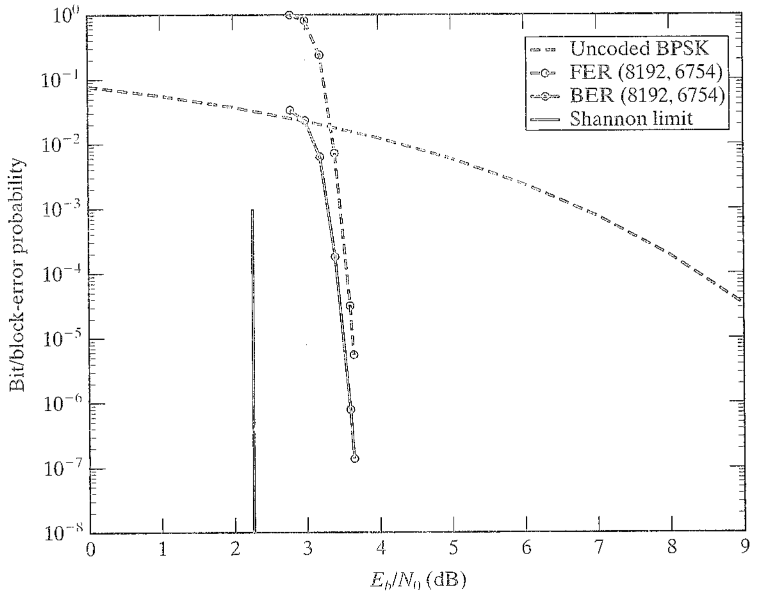

FIGURE 17.47: Error performance of the (8192, 6754) RS-based Gallager (6, 32)-regular LDPC code with construction field $GF(2^8)$ given in Example 17.42.

EXAMPLE 17.42

Suppose we use $GF(2^8)$ as the construction field and the (32, 2, 31) shortened RS code over $GF(2^8)$ as the base code. We set $\gamma = 6$. Then, the RS-based Gallager-LDPC code is a (8192, 6754) code with rate 0.824 and a minimum distance of at least 8. Its error performance with SPA decoding is shown in Figure 17.47. At a BER of 10^{-6}, it performs 1.25 dB from the Shannon limit and achieves a 6.7 dB gain over the uncoded BPSK.

The foregoing algebraic construction of LDPC codes is simple and yet very powerful. It gives a large class of regular LDPC codes. Codes in this class can be decoded with the SPA, weighted BF, BF, or one-step majority-logic decodings to provide a wide range of trade-offs between error performance and decoding complexity.

The matrix given by (17.99) is an array of permutation matrices that is in exactly the same form as the matrix given by (17.72). Hence, it can be masked to generate new LDPC codes.

17.19 CONCATENATIONS WITH LDPC AND TURBO CODES

In most applications of concatenated coding for error control, RS codes are used as the outer codes, and they are, in general, decoded with an algebraic decoding algorithm, such as the Berlekamp or Euclidean algorithm presented in Chapter 7. If a

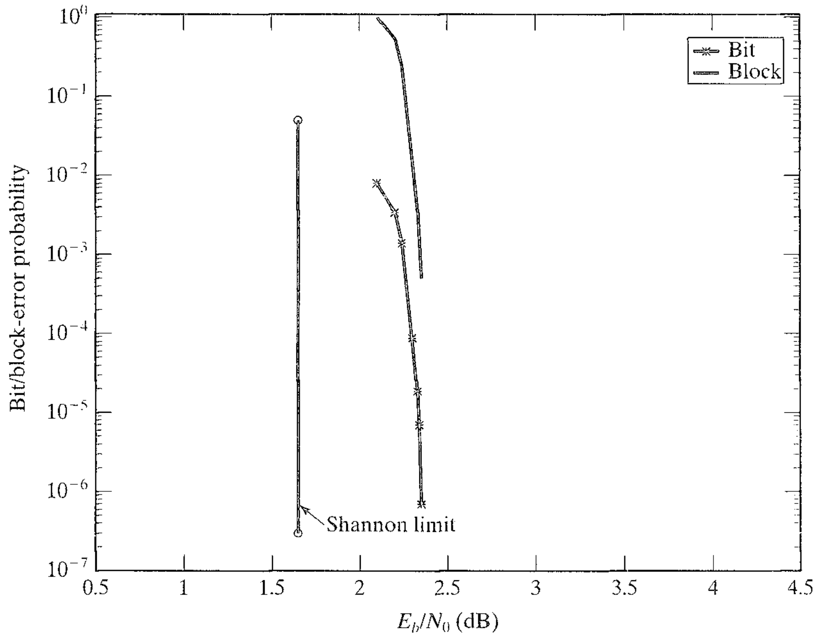

FIGURE 17.48: Bit- and block-error performance of a concatenated LDPC-turbo coding system with a turbo inner code and an extended EG-LDPC outer code.

significant improvement in error performance of a concatenated coding system with an RS code as the outer code is to be achieved, the RS outer code must be reasonably long and decoded with a sophisticated soft-decision decoding scheme that provides either optimal MLD performance or a suboptimal error performance. Unfortunately, the complexity of such a soft-decision decoding scheme or algorithm would be enormously large, and the decoder would be practically impossible to implement.

In this chapter we have shown that long high-rate finite-geometry LDPC codes with large minimum distances can be easily constructed and can be practically decoded with SPA decoding. They achieve very good error performance, especially the block-error performance. If such an LDPC code is used as the outer code in a concatenated coding system with a simple turbo code as the inner code, extremely good error performance and large coding gain should be achievable with practical implementation. With an LDPC code as the outer code and decoded with SPA decoding, the soft-output information provided by the inner turbo decoder can be fully utilized. We give an example to demonstrate the strength of such a combination of two powerful coding systems.

Consider a concatenated coding system in which the inner code is a high-rate block turbo code with the (64, 57) distance-4 Hamming code as the two constituent codes, and the outer code is the (65520, 61425) extended EG-LDPC code given in Example 17.14 [17]. The overall rate of this system is 0.75. The bit- and block-error performances of this concatenated LDPC-turbo coding system are shown in Figure 17.48. We see that this system achieves extremely good waterfall error performance. To achieve a BER of 10^{-6}, it requires an SNR of 2.35 dB, and at this BER, it performs only 0.7 dB away from the Shannon limit. This system is far superior to the concatenated coding scheme used in the NASA TDRS System presented in Section 15.6, whose overall rate is only 0.437.

Another form of concatenation of LDPC and turbo codes is to use an LDPC code as the two constituent codes in a parallel turbo coding arrangement, as described in Chapter 16. Because decoding of an LDPC code with the SPA is not trellis-based, a long LDPC code with large distance can be used as the constituent codes to achieve very good error performance without an error floor (or with an error floor at a very low error rate).

PROBLEMS

17.1 Does the following matrix satisfy the conditions of the parity-check matrix of an LDPC code given by Definition 17.1? Determine the rank of this matrix and give the codewords of its null space.

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

17.2 Form the transpose $\mathbb{H}^T$ of the parity-check matrix $\mathbb{H}$ given in Problem 17.1. Is $\mathbb{H}^T$ a low-density parity-check matrix? Determine the rank of $\mathbb{H}^T$ and construct the code given by the null space of $\mathbb{H}^T$.

17.3 Prove that the $(n, 1)$ repetition code is an LDPC code. Construct a low-density parity-check matrix for this code.

17.4 Consider the matrix $\mathbb{H}$ whose columns are all the m-tuples of weight 2. Does $\mathbb{H}$ satisfy the conditions of the parity-check matrix of an LDPC code? Determine the rank of $\mathbb{H}$ and its null space.

17.5 The following matrix is a low-density parity-check matrix. Determine the LDPC code given by the null space of this matrix. What is the minimum distance of this code?

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

17.6 Prove that the maximum-length code of length $2^m - 1$ presented in Section 8.3 is an LDPC code.

17.7 Construct the Tanner graph of the code given in Problem 17.1. Is the Tanner graph of this code acyclic? Justify your answer.

17.8 Construct the Tanner graph of the code given in Problem 17.2. Is the Tanner graph of this code acyclic? Justify your answer.

17.9 Construct the Tanner graph of the code given by the null space of the parity-check matrix given in Problem 17.5. Does the Tanner graph of this code contains cycles of length 6? Determine the number of cycles of length 6 in the graph.

17.10 Determine the orthogonal check-sums for every code bit of the LDPC code given by the null space of the parity-check matrix of Problem 17.5.

17.11 Prove that the minimum distance of the Gallager-LDPC code given in Example 17.2 is 6.

17.12 Determine the generator polynomial of the two-dimensional type-I (0, 3)th-order cyclic EG-LDPC code constructed based on the two-dimensional Euclidean geometry EG$(2, 2^3)$.

17.13 Determine the parameters of the parity-check matrix of the three-dimensional type-I (0,2)th-order cyclic EG-LDPC code $C_{EG,c}^{(1)}(3, 0, 2)$. Determine the generator polynomial of this code. What are the parameters of this code?

17.14 Determine the parameters of the companion code of the EG-LDPC code given in Problem 17.13.

17.15 Decode the two-dimensional type-I (0, 3)th-order cyclic EG-LDPC code with one-step majority-logic decoding and give the bit- and block-error performance for the AWGN channel with BPSK signaling.

17.16 Repeat Problem 17.15 with BF decoding.

17.17 Repeat Problem 17.15 with weighted majority-logic decoding.

17.18 Repeat Problem 17.15 with weighted BF decoding.

17.19 Repeat Problem 17.15 with SPA decoding.

17.20 Decode the three-dimensional type-II (0, 2)th-order quasi-cyclic EG-LDPC code given in Problem 17.14 with SPA decoding, and give the bit- and block-error performance of the code for the AWGN channel with BPSK signaling.

17.21 Consider the parity-check matrix $\mathbb{H}_{EG,c}^{(1)}$ of the three-dimensional type-I (0, 2)th-order cyclic EG-LDPC code given in Problem 17.13. Split each column of this parity-check matrix into five columns with rotating weight distribution. The result is a new low-density parity-check matrix that gives an extended EG-LDPC code. Decode this code with SPA decoding and give its bit- and block-error performances.

17.22 Construct a parity-check matrix of the Gallager-LDPC code with the following parameters: $m = 6$, $\rho = 4$, and $\gamma = 3$. Choose column permutations for the submatrices such that λ is no greater than 1.

17.23 Prove that the Tanner graph of a finite-geometry LDPC code contains cycles of length 6. Enumerate the number of cycles of length 6.

17.24 Prove that the minimum distance of an EG-Gallager LDPC code must be even. Use the result to prove the lower bound on minimum distance given by (17.68).

17.25 Construct an EG-Gallager LDPC code using six parallel bundles of lines in the two-dimensional Euclidean geometry EG$(2,2^5)$ over $GF(2^5)$. Compute its bit- and block-error performances with SPA decoding.

17.26 Construct a masked EG-Gallager LDPC code of length 1024 by decomposing the incidence matrices of eight parallel bundles of lines in EG$(2, 2^5)$ into 32×32 permutation matrices. To construct such a code, set $\rho = 32$, and form an 8×32 masking matrix with column and row weights 4 and 16, respectively, using four primitive 8-tuples over $GF(2)$. Compute the bit- and block-error performances using SPA decoding.

17.27 The incidence vectors of the lines in EG$(2, 2^5)$ not passing through the origin form a single 1023×1023 circulant G with weight 32. Construct a rate-1/2 quasi-cyclic code of length 8184 by decomposing G into a 4×8 array of 1023×1023 circulant permutation matrices. Compute the bit- and block-error performance of the code with SPA decoding.

17.28 Prove that there exist a primitive element α in $GF(241)$ and an odd positive integer c less than 241 such that $\alpha^{64} + 1 = \alpha^c$.

17.29 Design a concatenated turbo coding system with a finite-geometry LDPC code of your choice as the outer code. Construct the inner turbo code by using the second-order (32, 16) RM code as the component code. Give the bit-error performance of your designed system.

BIBLIOGRAPHY

1. R. G. Gallager, "Low Density Parity Check Codes," *IRE Trans. Inform. Theory*, IT-8: 21–28, January 1962.

2. R. G. Gallager, *Low Density Parity Check Codes*, MIT Press, Cambridge, 1963.

3. R. M. Tanner, "A Recursive Approach to Low Complexity Codes,"*IEEE Trans. Inform. Theory*, IT-27: 533–47, September 1981.

4. D. J. C. MacKay and R. M. Neal, "Near Shannon Limit Performance of Low Density Parity Check Codes," *Electron. Lett.*, 32 (18): 1645–46, 1996.

5. M. Sipser and D. Spielman, "Expander Codes," *IEEE Trans. Inform. Theory*, 42 (6): 1710–22, November 1996.

6. D. Spielman, "Linear-Time Encodable Error-Correcting Codes," *IEEE Trans. Inform. Theory*, 42 (6): 1723–31, November 1996.

7. M. C. Davey and D. J. C. MacKay, "Low Density Parity Check Codes over *GF*(q)," *IEEE Commun. Lett.*, 2(6), 165–67, June 1998.

8. M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved Low-Density Parity-Check Codes Using Irregular Graphs and Belief Propagation," *Proc. 1998 IEEE Intl. Symp. Inform. Theory*, p. 171, Cambridge, August 16–21, 1998.

9. D. J. C. Mackay, "Gallager Codes That Are Better Than Turbo Codes," *Proc. 36th Allerton Conf. Commun., Control, and Computing*, Monticello, Ill., September 1998.

10. ———, "Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. Inform. Theory*, 45 (2): 399–432, March 1999.

11. ———, "Sparse Graph Codes," *Proc. 5th Intl. Symp. Commun. Theory and Applications*, pp. 2–4, Ambleside, UK, July 11–16, 1999.

12. T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of Capacity-Approaching Irregular Codes," *IEEE Trans. Inform. Theory*, 47 (2): 619–37, February 2001.

13. T. Richardson and R. Urbanke, "The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding," *IEEE Trans. Inform. Theory*, 47: 599–618, February 2001.

14. S. Y. Chung, G. D. Forney, Jr., T. J. Richardson, and R. Urbanke, "On the Design of Low Density Parity Check Codes within 0.0045 dB of the Shannon Limit," *IEEE Commun. Lett.*, 5: 58–60, February 2001.

15. Y. Kou, S. Lin, and M. Fossorier, "Low Density Parity Check Codes Based on Finite Geometries: A Rediscovery," *Proc. IEEE Intl. Symp. Inform. Theory*, Sorrento, Italy, June 25–30, 2000.

16. S. Lin and Y. Kou, "A Geometric Approach to the Construction of Low Density Parity Check Codes," presented at the IEEE 29th Communication Theory Workshop, Haines City, Fla., May 7–10, 2000.

17. Y. Kou, S. Lin, and M. Fossorier, "Low Density Parity Check Codes Based on Finite Geometries: A Rediscovery and More," *IEEE Trans. Inform. Theory*, 47 (6): 2711–36, November 2001.

18. Y. Kou, S. Lin, and M. Fossorier, "Construction of Low Density Parity Check Codes—A Geometric Approach," *Proc. 2nd Intl. Symp. Turbo Codes and Related Topics*, Brest, France, September 4–7, 2000.

19. S. Lin, Y. Kou, and M. Fossorier, "Low Density Parity Check Codes Construction Based on Finite Geometries," *Proc. GLOBECOM 2000*, San Francisco, Calif., November 27–December 1, 2000.

20. S. Lin, Y. Kou, and M. Fossorier, "Finite Geometry Low Density Parity Check Codes: Construction, Structure and Decoding," *Proceedings of the ForneyFest*, Kluwer Academic, Boston, Mass., 2000.

21. A. Ventura and M. Chiani, "Design and Evaluation of Some High-Rate Low-Density Parity-Check Codes," *Proc. 2001 IEEE GlobeCom*, vol. 2, pp. 990–94, Nov. 25–29, 2001.

22. M. Yang, Y. Li, and W. E. Ryan, "Design of Efficiently Encodable Moderate-Length High-Rate Irregular LDPC Codes," to appear in *IEEE Trans. Commun.*, 2004.

23. J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, Calif., 1988.

24. S. L. Lauritzen and D. J. Spiegelhalter, "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems," *J. R. Stat. Soc. B*, 50: 157–224, 1988.

25. N. Wiberg, H.-A. Loeliger, and R. Kötter, "Codes and Iterative Decoding on General Graphs," *Eur. Trans. Telecommun.*, 6: 513–26, 1955.

26. R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo Decoding As an Instance of Pearl's Belief Propagation Algorithm," *IEEE J. Select. Areas Commun.*, 16: 140–52, February 1998.

27. F. R. Kschischang and B. J. Frey, "Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models," *IEEE J. Select. Areas Commun.*, 16 (12): 219–30, February 1998.

28. F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. Inform. Theory*, 47: 498–519, February 2001.

29. M. Fossorier, M. Mihaljevic, and H. Imai, "Reduced Complexity Iterative Decoding of Low Density Parity Check Codes," *IEEE Trans. Commun.*, 47: 673–80, May 1999.

30. R. Lucas, M. Fossorier, Y. Kou, and S. Lin, "Iterative Decoding of One-Step Majority Logic Decodable Codes Based on Belief Propagation," *IEEE Trans. Commun.*, 48 (6): 931–37, June 2000.

31. C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," *Proc. 1993 IEEE Intl. Conf. Commun.*, pp. 1064–70, Geneva, Switzerland, May 1993.

32. C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Trans. Commun.*, 44: 1261–71, October 1996.

33. S. Benedetto and G. Montorsi, "Unveiling Turbo Codes: Some Results on Parallel Concatenated Coding Schemes," *IEEE Trans. Inform. Theory*, 42 (2): 409–28, March 1996.

34. J. Hagenauer, E. Offer, and L. Papke, "Iterative Decoding of Binary Block and Convolutional Codes," *IEEE Trans. Inform. Theory*, 42: 429–45, March 1996.

35. N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, Prentice Hall, Englewood Cliffs, N.J., 1974.

36. S. M. Aji and R. J. McEliece, "The Generalized Distributive Law," *IEEE Trans. Inform. Theory,* 46 (2): 325–43, March 2000.

37. N. Wiberg, "Codes and Decoding on General Graphs," Ph.D. Diss., Dept. of Electrical Engineering, University of Linköping, Linköping, Sweden, April 1996

38. T. Etzion, A. Trachtenberg, and A. Vardy, "Which Codes Have Cycle-Free Tanner Graphs," *IEEE Trans. Inform. Theory*, 45 (6): 2173–81, September 1999.

39. G. D. Forney, Jr., "Codes on Graphs: A Survey for Algebraists," *Proc. 13th Intl. Symp. Applied Algebra, Algebraic Algorithms and Error-Correcting Codes,* Honolulu, November 15–19, 1999.

40. R. D. Carmichael, *Introduction to the Theory of Groups of Finite Order,* Dover, New York, 1956.

41. H. B. Mann, *Analysis and Design of Experiments,* Dover, New York, 1949.

42. T. Kasami, S. Lin, and W. W. Peterson, "Polynomial Codes," *IEEE Trans. Inform. Theory,* 14: 807–14, 1968.

43. R. C. Bose and D. J. Ray-Chaudhuri, "On a Class of Error Correcting Binary Group Codes," *Inform. Control*, no. 3: 68–79, March 1960.

44. W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes,* 2d ed., MIT Press, Cambridge, 1972.

45. E. R. Berlekamp, *Algebraic Coding Theory,* McGraw-Hill, New York, 1968.

46. S. Lin, "On the Number of Information Symbols in Polynomial Codes," *IEEE Trans. Inform. Theory,* IT-18: 785–94, November 1972.

47. E. J. Weldon, Jr., "Difference-Set Cyclic Codes," *Bell Syst. Tech. J.*, 45: 1045–55, September 1966.

48. V. D. Kolesnik, "Probability Decoding of Majority Codes," *Probl. Peredachi Inform.*, 7: 3–12, July 1971.

49. R. Lucas, M. Bossert, and M. Breitback, "On Iterative Soft-Decision Decoding of Linear Block Codes and Product Codes," *IEEE J. Select. Areas Commun.*, 16: 276–96, February 1998.

50. S. Lin "Shortened Finite Geometry Codes," *IEEE Trans. Inform. Theory*, IT-18 (5): 692–96, September 1972.

51. H. Tang, J. Xu, Y. Kou, S. Lin, and K. Abdel-Ghaffar, "On Algebraic Construction of Low Density Parity Check Codes," *Proc. 2002 IEEE Intl. Symp. Inform. Theory,* p. 482, Lausanne, Switzerland, June 30–July 5, 2002.

52. H. Tang, J. Xu, Y. Kou, S. Lin, and K. Abdel-Ghaffar, "On Algebraic Construction of Gallager and Circulant Low Density Parity Check Codes," to appear in *IEEE Trans. Inform. Theory*, June 2004.

53. J. Xu, L. Chen, I. Djurdjevic, S. Lin, and K. Abdel-Ghaffar, "Construction of Regular and Irregular Low Density Parity Check Codes, Based on Geometry Decomposition and Masking," submitted to *IEEE Trans. Inform. Theory*, 2004.

54. S. Liu, L. Chen, J. Xu, and I. Djurdjevic, "Near Shannon Limit Quasi-Cyclic Low-Density Parity-Check Codes," *Proc. IEEE GLOBECOM 2003*, San Francisco, Calif., December 1–5, 2003.

55. L. Chen, J. Xu, I. Djurdjevic, and S. Lin, "Near Shannon Limit Quasi-Cyclic Low-Density Parity-Check Codes," to appear in *IEEE Trans. Commun.*, July 2004.

56. Y. Kou, J. Xu, H. Tang, S. Lin, and K. Abdel-Ghaffar, "On Circulant Low Density Parity Check Codes," *Proc. IEEE Intl. Symp. Inform. Theory*, p. 200, Lausanne, Switzerland, June 30–July 5, 2002.

57. C. L. Liu, *Introduction to Combinatorial Mathematics*, McGraw-Hill, New York, 1968.

58. C. J. Colbourn and H. J. Dinitz, *The CRC Handbook of Combinatorial Designs*, Boca Raton, Fla., CRC Publications, 1996.

59. I. Djurdjevic, S. Lin, and K. Abdel-Ghaffar, "On a Graph-Theoretic Construction of Low Density Parity Check Codes," *IEEE Commun. Lett.*, 7: 171–73, April 2003.

60. J. Rosenthal and P. O Vontobel, "Construction of Regular and Irregular LDPC Codes Using Ramamujan Graphs and Ideas from Margulis," *Proc. 2001 IEEE Intl. Symp. Inform. Theory*, Washington, D.C., p. 4, June 24–29, 2001.

61. P. O Vontobel and R. M. Tanner, "Construction of Codes Based on Finite Generalized Quadrangles for Iterative Decoding," *Proc. 2001 IEEE Intl. Symp. Inform. Theory*, Washington, D.C., p. 223, June 24–29, 2001.

62. P. O Vontobel and H. A. Loeliger, "Irregular Codes from Regular Graphs," *Proc. 2002 IEEE Intl. Symp. Inform. Theory*, Lausanne, Switzerland, p. 284, June 30–July 4, 2002.

63. R. C. Bose, "On the Construction of Balanced Incomplete Block Designs," *Ann. Eugenics* 9: 353–99, 1939.

64. B. Ammar, B. Honary, Y. Kou, J. Xu, and S. Lin, "Construction of Low Density Parity Check Codes Based on Balanced Incomplete Block Designs," to appear in *IEEE Trans. Inform. Theory*, June 2004.

65. B. Vasic and O. Milenkovic, "Combinatorial Construction of Low-Density Parity Check Codes for Iterative Decoding," to appear in *IEEE Trans. Inform. Theory*, 2004.

66. I. Djurdjevic, J. Xu, K. Abdel-Ghaffar, and S. Lin, "Construction of Low-Density Parity-Check Codes Based on Shortened Reed–Solomon Codes with Two Information Symbols," *IEEE Commun. Lett.*, no. 7: 317–19, July 2003.