Chapter 12
LDPC Codes

12.1 Background and Notation

LDPC codes are linear block codes whose parity-check matrix—as the name implies—is sparse. These codes can be iteratively decoded using the sum product [9] or equivalently the belief propagation [24] soft decision decoder. It has been shown, for example by Chung et al. [3], that for long block lengths, the performance of LDPC codes is close to the channel capacity. The theory of LDPC codes is related to a branch of mathematics called graph theory. Some basic definitions used in graph theory are briefly introduced as follows.

Definition 12.1 (*Vertex, Edge, Adjacent and Incident*) A graph, denoted by $G(V, E)$, consists of an ordered set of vertices and edges.

- **(Vertex)** A vertex is commonly drawn as a node or a dot. The set $V(G)$ consists of vertices of $G(V, E)$ and if v is a vertex of $G(V, E)$, it is denoted as $v \in V(G)$. The number of vertices of $V(G)$ is denoted by $|V(G)|$.
- **(Edge)** An edge (u, v) *connects* two vertices $u \in V(G)$ and $v \in V(G)$ and it is drawn as a line connecting vertices u and v. The set $E(G)$ contains pairs of elements of $V(G)$, i.e. $\{(u, v) \mid u, v \in V(G)\}$.
- **(Adjacent and Incident)** If $(u, v) \in E(G)$, then $u \in V(G)$ and $v \in V(G)$ are adjacent or neighbouring vertices of $G(V, E)$. Similarly, the vertices u and v are incident with the edge (u, v).

Definition 12.2 (*Degree*) The degree of a vertex $v \in V(G)$ is the number of edges that are incident with vertex v, i.e. the number of edges that are connected to vertex v.

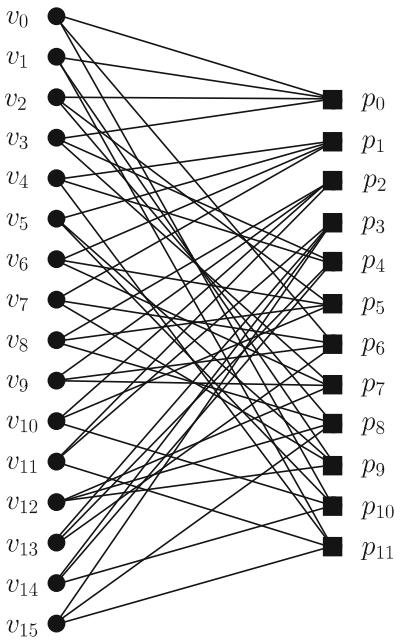
Definition 12.3 (*Bipartite or Tanner graph*) Bipartite or Tanner graph $G(V, E)$ consists of two disjoint sets of vertices, say $V_v(G)$ and $V_p(G)$, such that $V(G) = V_v(G) \cup V_p(G)$, and every edge $(v_i, p_j) \in E(G)$, such that $v_i \in V_v(G)$ and $p_j \in V_p(G)$ for some integers i and j.

M. Tomlinson et al., *Error-Correction Coding and Decoding*,
Signals and Communication Technology,
DOI 10.1007/978-3-319-51103-0_12

An $[n, k, d]$ LDPC code may be represented by a Tanner graph $G(V, E)$. The parity-check matrix $\boldsymbol{H}$ of the LDPC code consists of $|V_p(G)| = n - k$ rows and $|V_v(G)| = n$ columns. The set of vertices $V_v(G)$ and $V_p(G)$ are called *variable* and *parity-check* vertices, respectively. Figure 12.1 shows the parity check and the cor-

$$\boldsymbol{H} = \begin{array}{c} \begin{matrix} v_0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 & v_9 & v_{10} & v_{11} & v_{12} & v_{13} & v_{14} & v_{15} \end{matrix} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{matrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \\ p_{10} \\ p_{11} \end{matrix} \end{array}$$

(a) Parity check matrix



(b) Tanner graph

Fig. 12.1 Representations of a [16, 4, 4] LDPC code

responding Tanner graph of a [16, 4, 4] LDPC code. Let $V_v(G) = (v_0, v_1, \ldots, v_{n-1})$ and $V_p(G) = (p_0, p_1, \ldots, p_{n-k-1})$; we can see that for each $(v_i, p_j) \in E(G)$, the ith column and jth row of $\boldsymbol{H}$, $H_{j,i} \neq 0$, for $0 \leq i \leq n-1$ and $0 \leq j \leq n-k-1$.

Definition 12.4 (*Cycle*) A cycle in a graph $G(V, E)$ is a sequence of distinct vertices that starts and ends in the same vertex. For bipartite graph $G(V, E)$, exactly half of these distinct vertices belong to $V_v(G)$ and the remaining half belong to $V_p(G)$.

Definition 12.5 (*Girth and Local Girth*) The girth of graph $G(V, E)$ is the length of the shortest cycle in the graph $G(V, E)$. The local girth of a vertex $v \in V(G)$ is the length of shortest cycle that passes through vertex v.

The performance of a typical iteratively decodable code (e.g. an LDPC or turbo code) may be partitioned into three regions, namely erroneous, waterfall and error floor regions, see Fig. 12.2. The erroneous region occurs at low E_b/N_0 values and is indicated by the inability of the iterative decoder to correctly decode almost all of the transmitted messages. As we increase the signal power, the error rate of the iterative decoder decreases rapidly—resembling a waterfall. The E_b/N_0 value at which the waterfall region starts is commonly known as the *convergence threshold* in the literature. At higher E_b/N_0 values, the error rate starts to flatten—introducing an error floor in the frame error rate (FER) curve.

In addition to this FER curve, the offset sphere packing lower bound and the probability of error based on the union bound argument as described in Chap. 1 are also plotted in Fig. 12.2. The sphere packing lower bound represents the region of

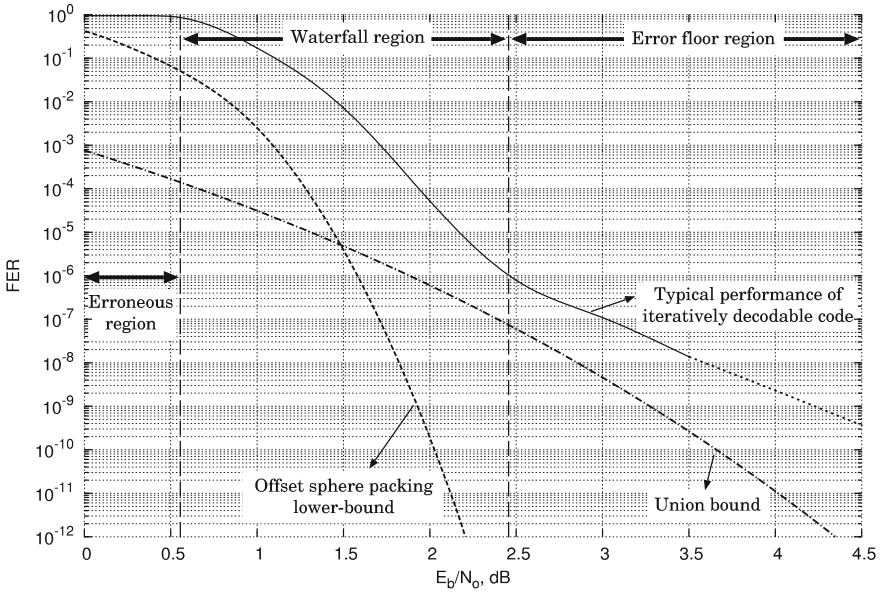


Fig. 12.2 Waterfall and error regions of a typical LDPC code for the AWGN channel

attainable performance of a coding system. The performance to the left of this lower bound is not attainable, whereas that to the right may be achieved by some coding and decoding arrangements. The other curve is the union bound of the probability of error, which is dominated by the low Hamming weight codewords and the number of codewords of these Hamming weights. The larger the minimum Hamming distance of a code, the lower the union bound typically. For iteratively decodable codes which are not designed to maximise the minimum Hamming distance, the union bound intersects with the offset sphere packing lower bound at relatively low E_b/N_0 values.

It may be seen that, with an ideal soft decision decoder, the performance of a coding system would follow the sphere packing lower bound and at higher E_b/N_0 values, the performance floors due to the limitation of the minimum Hamming weight codewords. However, as depicted in Fig. 12.2, there is a relatively wide gap between the union bound and the error floor of a typical iteratively decodable code. This is an inherent behaviour of iteratively decodable codes and it is attributed to the weakness of the iterative decoder. There are other error events, which are not caused by the minimum Hamming weight codewords, that prevent the iterative decoder from reaching the union bound.

In terms of the construction technique, we may divide LDPC codes into two categories: random and algebraic LDPC codes. We may also classify LDPC codes into two categories depending on the structure of the parity-check matrix, namely regular and irregular codes—refer to Sect. 12.1.1 for the definition. Another attractive construction method that has been shown to offer capacity-achieving performance is non-binary construction.

12.1.1 Random Constructions

Gallager [8] introduced the (n, λ, ρ) LDPC codes where n represents the block length whilst the number of non-zeros per column and the number of non-zeros per row are represented by λ and ρ, respectively.

The short notation (λ, ρ) is also commonly used to represent these LDPC codes. The coderate of the Gallager (λ, ρ) codes is given by

$$R = 1 - \frac{\lambda}{\rho}.$$

An example of the parity-check matrix of a Gallager (λ, ρ) LDPC code is shown in Fig. 12.1a. It is a [16, 4, 4] code with a λ of 3 and a ρ of 4. The parity-check matrix of the (λ, ρ) Gallager codes always have a fixed number of non-zeros per column and per row, and because of this property, this class of LDPC codes is termed regular LDPC codes. The performance of the Gallager LDPC codes in the waterfall region is not as satisfactory as that of turbo codes for the same block length and code rate. Many efforts have been devoted to improve the performance of the LDPC codes and one example that provides significant improvement is the introduction of the irregular

LDPC codes by Luby et al. [18]. The irregular LDPC codes, as the name implies, do not have a fixed number of non-zeros per column or per row and thus the level of error protection varies over a codeword. The columns of a parity check matrix that have a higher number of non-zeros provide stronger error protection than those that have a lower number of non-zeros. Given an input block in iterative decoding, errors in the coordinates of this block, whose columns of the parity-check matrix have a larger number of non-zeros, will be corrected earlier, i.e. only a small number of iterations are required. In the subsequent iterations, the corrected values in these coordinates will then be utilised to correct errors in the remaining coordinates of the block.

Definition 12.6 (*Degree Sequences*) The polynomial $\Lambda_\lambda(x) = \sum_{i\geq 1} \lambda_i x^i$ is called the symbol or variable degree sequence, where λ_i is the fraction of vertices of degree i. Similarly, $\Lambda_\rho(x) = \sum_{i\geq 1} \rho_i x^i$ is the check degree sequence, where ρ_i is the fraction of vertices of degree i.

The degree sequences given in the above definition are usually known as *vertex-oriented degree sequences*. Another representations are *edge-oriented degree sequences* which consider the fraction of edges that are connected to a vertex of certain degree. Irregular LDPC codes are defined by these degree sequences and it is assumed that the degree sequences are vertex-oriented.

Example 12.1 An irregular LDPC code with the following degree sequences

$$\Lambda_\lambda(x) = 0.5x^2 + 0.26x^3 + 0.17x^5 + 0.07x^{10}$$
$$\Lambda_\rho(x) = 0.80x^{14} + 0.20x^{15}$$

has 50, 26, 17 and 7% of the columns with 2, 3, 5 and 10 ones per column, respectively, and 80 and 20% of the rows with 14 and 15 ones per row, respectively.

Various techniques have been proposed to design good degree distributions. Richardson et al. [27] used *density evolution* to determine the convergence threshold and to optimise the degree distributions. Chung et al. [4] simplified the density evolution approach using *Gaussian approximation*. With the optimised degree distributions, Chung et al. [3] showed that the bit error rate performance of a long block length ($n = 10^7$) irregular LDPC code was within 0.04 dB away from the capacity limit for binary transmission over the AWGN channel, discussed in Chap. 1. This is within 0.18 dB of Shannon's limit [30]. The density evolution and Gaussian approximation methods, which make use of the concentration theorem [28], can only be used to design the degree distributions for infinitely long LDPC codes. The concentration theorem states that the performance of cycle-free LDPC codes can be characterised by the average performance of the ensemble. The cycle-free assumption is only valid for infinitely long LDPC codes and cycles are inevitable for finite block-length LDPC codes. As may be expected, the performance of finite block-length LDPC codes with degree distributions derived based on the concentration theorem differs considerably from the ensemble performance. There are various techniques to design good finite

block-length LDPC codes, for instance see [1, 2, 10, 33]. In particular, the work of Hu et al. [10] with the introduction of the progressive edge-growth (PEG) algorithm to construct both regular and irregular LDPC codes, that of Tian et al. [33] with the introduction of *extrinsic message degree* and recently, that of Richter et al. [29] which improves the original PEG algorithm by introducing some construction constraints to avoid certain cycles involving variable vertices of degree 3, have provided significant contributions to the construction of practical LDPC codes as well as the lowering of the inherent error floor of these codes.

12.1.2 Algebraic Constructions

In general, LDPC codes constructed algebraically have a regular structure in their parity-check matrix. The algebraic LDPC codes offer many advantages over randomly generated codes. Some of these advantages are

1. The important property such as the minimum Hamming distance can be easily determined or in the worst case, lower and upper bounds may be mathematically derived. These bounds are generally more accurate than estimates for random codes.
2. The minimum Hamming distance of algebraic LDPC codes is typically higher than that of random codes. Due to the higher minimum Hamming distance, algebraic codes are not that likely to suffer from an early error floor.
3. The existence of a known structure in algebraic codes usually offers an attractive and simple encoding scheme. In the case of random codes, in order to carry out encoding, a Gaussian elimination process has to be carried out in the first place and the entire reduced echelon parity-check matrix has to be stored in the memory. Algebraically constructed codes such as cyclic or quasi-cyclic codes can be completely defined by polynomials. The encoding of cyclic or quasi-cyclic codes may be simply achieved using a linear-feedback shift-register circuit and the memory requirement is minimum. Various efficient techniques for encoding random LDPC codes have been proposed, see Ping et al. [26] for example, but none of these techniques simplifies the storage requirements. The simplicity of the encoder and decoder structure has led to many algebraically constructed LDPC codes being adopted as industry standards [5].
4. Cyclic LDPC codes have n low Hamming weight parity-check equations and therefore, compared to random codes, these cyclic LDPC codes have k extra equations for the iterative decoder to iterate with and this leads to improved performance.

One of the earliest algebraic LDPC code constructions was introduced by Margulis [21] using the Ramanujan graphs. Lucas et al. [19] showed that the well-known different set cyclic (DSC) [36] and one-step majority-logic decodable (OSMLD) [17] codes have good performance under iterative decoding. The iterative soft decision decoder offers significant improvement over the conventional

hard decision majority-logic decoder. Another class of algebraic codes is the class of the Euclidean and projective geometry codes which are discussed in detail by Kou et al. [16]. Other algebraic constructions include those that use combinatorial techniques [13–15, 35].

It has been observed that in general, there is an inverse performance relationship between the minimum Hamming distance of the code and the convergence of the iterative decoder. Irregular codes converge well with iterative decoding, but the minimum Hamming distance is relatively poor. In contrast, algebraically constructed LDPC codes, which have high minimum Hamming distance, tend not to converge well with iterative decoding. Consequently, compared to the performance of irregular codes, algebraic LDPC codes may perform worse in the low SNR region and perform better in the high SNR region. This is attributed to the early error floor of the irregular codes. As will be shown later, for short block lengths ($n < 350$), cyclic algebraic LDPC codes offer some of the best performance available.

12.1.3 Non-binary Constructions

LDPC codes may be easily extended so that the symbols take values from the finite-field $\mathbb{F}_{2^m}$ and Davey et al. [6] were the pioneers in this area. Given an LDPC code over $\mathbb{F}_2$ with parity-check matrix $\boldsymbol{H}$, we may construct an LDPC code over $\mathbb{F}_{2^m}$ for $m \geq 2$ by simply replacing every non-zero element of $\boldsymbol{H}$ with any non-zero element of $\mathbb{F}_{2^m}$ in a random or structured manner. Davey et al. [6] and Hu et al. [11] have shown that the performance of LDPC codes can be improved by going beyond the binary field. The non-binary LDPC codes have better convergence behaviour under iterative decoding. Using some irregular non-binary LDPC codes, whose parity-check matrices are derived by randomly replacing the non-zeros of the PEG-constructed irregular binary LDPC codes, Hu et al. [11] demonstrated that an additional coding gain of 0.25 dB was achieved. It may be regarded that the improved performance is attributable to the improved graph structure in the non-binary arrangement. Consider a cycle of length 6 in the Tanner graph of a binary LDPC code, which is represented as the following sequence of pairs of edges $\{(v_0, p_0), (v_3, p_0), (v_3, p_2), (v_4, p_2), (v_4, p_1), (v_0, p_1)\}$. If we replace the corresponding entries in the parity-check matrix with some non-zeros over $\mathbb{F}_{2^m}$ for $m \geq 2$, provided that these six entries are not all the same, the cycle length becomes longer than 6. According to McEliece et al. [22] and Etzion et al. [7], the non-convergence of the iterative decoder is caused by the existence of cycles in the Tanner graph representation of the code. Cycles, especially those of short lengths, introduce correlations of reliability information exchanged in iterative decoding. Since cycles are inevitable for finite block length codes, it is desirable to have LDPC codes with large girth.

The non-binary LDPC codes also offer an attractive matching for higher order modulation methods. The impact of increased complexity of the symbol-based iterative decoder can be moderated as the reliability information from the component

codes may be efficiently evaluated using the frequency-domain dual-code decoder based on the Fast Walsh–Hadamard transform [6].

12.2 Algebraic LDPC Codes

Based on idempotents and cyclotomic cosets, see Chap. 4, a class of cyclic codes that is suitable for iterative decoding may be constructed. This class of cyclic codes falls into the class of one-step majority-logic decodable (OSMLD) codes whose parity-check polynomial is orthogonal on each bit position—implying the absence of a girth of 4 in the underlying Tanner graph, and the corresponding parity-check matrix is sparse, and thus can be used as LDPC codes.

Definition 12.7 (*Binary Parity-Check Idempotent*) Let $\mathscr{M} \subseteq \mathscr{N}$ and let the polynomial $u(x) \in T(x)$ be defined by

$$u(x) = \sum_{s \in \mathscr{M}} e_s(x) \tag{12.1}$$

where $e_s(x)$ is an idempotent. The polynomial $u(x)$ is called a binary parity-check idempotent.

The binary parity-check idempotent $u(x)$ can be used to describe an $[n, k]$ cyclic code as discussed in Chap. 4. Since $\mathrm{GCD}(u(x), x^n - 1) = h(x)$, the polynomial $\bar{u}(x) = x^{\deg(u(x))} u(x^{-1})$ and its n cyclic shifts $\pmod{x^n - 1}$ can be used to define the parity-check matrix of a binary cyclic code. In general, $\mathrm{wt}_H(\bar{u}(x))$ is much lower than $\mathrm{wt}_H(h(x))$, and therefore a low-density parity-check matrix can be derived from $\bar{u}(x)$.

Let the parity-check polynomial $\bar{u}(x) = x^{\bar{u}_0} + x^{\bar{u}_1} + \cdots + x^{\bar{u}_t}$ of weight $t + 1$. Since the code defined by $\bar{u}(x)$ is cyclic, for each non-zero coefficient $\bar{u}_i$ in $\bar{u}(x)$, there are another t parity-check polynomials of weight $t + 1$, which also have a non-zero coefficient at position $\bar{u}_i$. Furthermore, consider the set of these $t + 1$ polynomials that have a non-zero coefficient at position $\bar{u}_i$, there is no more than one polynomial in the set that have a non-zero at position $\bar{u}_j$ for some integer j. In other words, if we count the number of times the positions $0, 1, \ldots, n - 1$ appear in the exponents of the aforementioned set of $t + 1$ polynomials, we shall find that all positions except $\bar{u}_i$ appear at most once. This set of $t + 1$ polynomials is said to be *orthogonal* on position $\bar{u}_i$. The mathematical expression of this orthogonality is given in the following definition and lemma.

Definition 12.8 (*Difference Enumerator Polynomial*) Let the polynomial $f(x) \in T(x)$. The difference enumerator of $f(x)$, denoted as $\mathscr{D}(f(x))$, is defined as

$$\mathscr{D}(f(x)) = f(x)\, f\left(x^{-1}\right) = d_0 + d_1 x + \cdots + d_{n-1} x^{n-1}, \tag{12.2}$$

where it is assumed that $\mathscr{D}(f(x))$ is a modulo $x^n - 1$ polynomial with coefficients taking values from $\mathbb{R}$ (real coefficients).

Lemma 12.1 *Let d_i for $0 \leq i \leq n-1$ denote the coefficients of $\mathscr{D}(\bar{u}(x))$. If $d_i \in \{0, 1\}$, for all $i \in \{1, 2, \ldots, n-1\}$, the parity-check polynomial derived from $\bar{u}(x)$ is orthogonal on each position in the n-tuple. Consequently,*

(i) the minimum distance of the resulting LDPC code is $1 + \mathrm{wt}_H(\bar{u}(x))$, and
(ii) the underlying Tanner Graph has girth of at least 6.

Proof (i) [25, Theorem 10.1] Let a codeword $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ and $c(x) \in T(x)$. For each non-zero bit position c_j of $c(x)$, where $j \in \{0, 1, \ldots, n-1\}$, there are $\mathrm{wt}_H(u(x))$ parity-check equations orthogonal to position c_j. Each of the parity-check equations must check another non-zero bit c_l, where $l \neq j$, so that the equation is satisfied. Clearly, $\mathrm{wt}_H(c(x))$ must equal to $1 + \mathrm{wt}_H(u(x))$ and this is the minimum weight of all codewords.
(ii) The direct consequence of having orthogonal parity-check equations is the absence of cycles of length 4 in the Tanner Graphs. Let a, b and c, where $a < b < c$, be three distinct coordinates in an n-tuple, since $d_i \in \{0, 1\}$ for $1 \leq i \leq n-1$, this implies that $b - a \neq c - b$. We know that $q(b-a) \pmod{n} \in \{1, 2, \ldots, n-1\}$ and thus, $q(b-a) \pmod{n} \equiv (c-b)$ for some integer $q \in \{1, 2, \ldots, n-1\}$. If we associate the integers a, b and c with some variable vertices in the Tanner graph, a cycle of length 6 is produced.

It can be deduced that the cyclic LDPC code with parity-check polynomial $\bar{u}(x)$ is an OSMLD code if $d_i \in \{0, 1\}$, for all $i \in \{1, 2, \ldots, n-1\}$ or a difference set cyclic (DSC) code if $d_i = 1$, for all $i \in \{1, 2, \ldots, n-1\}$, where d_i is the coefficient of $\mathscr{D}(\bar{u}(x))$.

In order to arrive at either OSMLD or DSC codes, the following design conditions are imposed on $\bar{u}(x)$ and therefore, $u(x)$:

Condition 12.1 The idempotent $u(x)$ must be chosen such that

$$\mathrm{wt}_H(u(x))\,(\mathrm{wt}_H(u(x)) - 1)) \leq n - 1.$$

Proof There are $\mathrm{wt}_H(u(x))$ polynomials of weight $\mathrm{wt}_H(u(x))$ that are orthogonal on position j for some integer j. The number of distinct positions in this set of polynomials is $\mathrm{wt}_H(u(x))\,(\mathrm{wt}_H(u(x)) - 1)$, and this number must be less than or equal to the total number of distinct integers between 1 and $n-1$.

Condition 12.2 Following Definition 12.8, let $W = \{i \mid d_i = 1,\ 1 \leq i \leq n-1\}$, the cardinality of W must be equal to $\mathrm{wt}_H(u(x))\,(\mathrm{wt}_H(u(x)) - 1)$.

Proof The cyclic differences between the exponents of polynomial $u(x)$ are given by $\mathscr{D}(u(x)) = \sum_{i=0}^{n-1} d_i x^i$, where the coefficient d_i is the number of differences and the exponent i is the difference. The polynomial $u(x)$ and some of its cyclic shifts are orthogonal on position 0 and this means that all of the cyclic differences

between the exponents of $u(x)$ (excluding zero) must be distinct, i.e. $d_i \in \{0, 1\}$ for $1 \leq i \neq n-1$. Since the weight of $u(x)$ excluding x^0 is $\mathrm{wt}_H(u(x)) - 1$ and there are $\mathrm{wt}_H(u(x))$ cyclic shifts of $u(x)$ that are orthogonal to x^0, the number of distinct exponents in the cyclic differences is $\mathrm{wt}_H(u(x))(\mathrm{wt}_H(u(x)) - 1) = W$.

Condition 12.3 The exponents of $u(x)$ must not contain a common factor of n, otherwise a degenerate code, a repetition of a shorter cyclic code, is the result.

Proof If the exponents of $u(x)$ contain a common factor of n, p with $n = pr$, then factors of $u(x)$ divide $x^r - 1$ and form a cyclic code of length r. Every codeword of the longer code is a repetition of the shorter cyclic code.

Condition 12.4 Following (12.1), unless $\mathrm{wt}_H(e_s(x)) = 2$, the binary parity-check idempotent $e_s(x)$ must not be self-reciprocal, i.e. $e_s(x) \neq e_i\left(x^{-1}\right)$, for all $i \in \mathscr{M}$.

Proof The number of non-zero coefficients of $\mathscr{D}(e_s(x))$ is equal to

$$\mathrm{wt}_H(e_s(x))\,(\mathrm{wt}_H(e_s(x)) - 1)\,.$$

For a self-reciprocal case, $e_s(x)e_s\left(x^{-1}\right) = e_s^2(x) = e_s(x)$ with $\mathrm{wt}_H(e_s(x))$ non-zero coefficients. Following Condition 12.1, the inequality

$$\mathrm{wt}_H(e_s(x))\,(\mathrm{wt}_H(e_s(x) - 1) \leq \mathrm{wt}_H(e_s(x))$$

becomes equality if and only if $\mathrm{wt}_H(e_s(x)) = 2$.

Condition 12.5 Following (12.1), $u(x)$ must not contain $e_s\left(x^{-1}\right)$, for all $i \in \mathscr{M}$, unless $e_s(x)$ is self-reciprocal.

Proof If $u(x)$ contains $e_s\left(x^{-1}\right)$ for $i \in \mathscr{M}$, then $\mathscr{D}(u(x))$ will contain both $e_s(x)e_s\left(x^{-1}\right)$ and $e_s\left(x^{-1}\right)e_s(x)$, hence, some of the coefficients of $\mathscr{D}(e_s(x))$, $d_i \neq \{0, 1\}$ for some integer i.

Although the above conditions seem overly restrictive, they turn out to be helpful in code construction. Codes may be designed in stage-by-stage by adding candidate idempotents to $u(x)$, checking the above conditions at each stage.

In order to encode the cyclic LDPC codes constructed, there is no need to determine $g(x)$. With α defined as a primitive n^{th} root of unity, it follows from Lemma 4.4 that $u(\alpha^i) \in \{0, 1\}$ for $0 \leq i \leq n-1$. Let $\mathscr{J} = (j_0, j_1, \ldots, j_{n-k-1})$ be a set of integers between 0 and $n-1$, such that $g(\alpha^j) = 0$, for all $j \in \mathscr{J}$. Because $u(x)$ does not contain α^j as its roots, it follows that $u(\alpha^j) = 1$, for all $j \in \mathscr{J}$. In $\mathbb{F}_2$, $1 + u(\alpha^j) = 0$ and the polynomial $1 + u(x) = e_g(x)$, the generating idempotent of the code may be used to generate the codewords as an alternative to $g(x)$.

The number of information symbols of the cyclic LDPC codes can be determined either from the number of roots of $u(x)$ which are also roots of unity, i.e. $n - \mathrm{wt}_H(U(z))$, or from the degree of $(u(x), x^n - 1) = h(x)$.

Example 12.2 Consider the design of a cyclic LDPC code of length 63. The cyclotomic coset modulo 63 is given in Example 4.2. Let $u(x)$ be defined by C_9, i.e. $u(x) = e_9(x) = x^9(1 + x^9 + x^{27})$. $\mathscr{D}(\bar{u}(x))$ indicates that the parity-check matrix defined by $\bar{u}(x)$ has no cycles of length 4; however, following Condition 12.3, it is a degenerate code consisting of repetitions of codewords of length 7.

With $u(x) = e_{23}(x) = x^{23}(1 + x^6 + x^{20} + x^{23} + x^{30} + x^{35})$, the resulting cyclic code is a [63, 31, 6] LDPC code which is non-degenerate and its underlying Tanner graph has girth of 6. This code can be further improved by adding $e_{21}(x)$ to $u(x)$. Despite $e_{21}(x)$ being self-reciprocal, its weight is 2 satisfying Condition 12.4. Now, $u(x) = x^{21}(1 + x^2 + x^8 + x^{21} + x^{22} + x^{25} + x^{32} + x^{37})$, and it is a [63, 37, 9] cyclic LDPC code.

Based on the theory described above, an algorithm which exhaustively searches for all non-degenerate cyclic LDPC codes of length n which have orthogonal parity-check polynomials has been developed, and it is given in Algorithm 12.1.

Algorithm 12.1 CodeSearch($\mathbf{V}$, $index$)

Input:

$index \Leftarrow$ an integer that is initialised to -1
$\mathbf{V} \Leftarrow$ a vector that is initialised to $\emptyset$
$\mathscr{S} \Leftarrow \mathscr{N}$ excluding 0

Output:

CodesList contains set of cyclic codes which have orthogonal parity-check polynomial

1: $\mathbf{T} \Leftarrow \mathbf{V}$
2: **for** $(i{=}index{+}1; i \leq |\mathscr{S}|; i{+}{+})$ **do**
3: $\quad \mathbf{T}_{\text{prev}} \Leftarrow \mathbf{T}$
4: $\quad$ **if** $\left(\sum_{\forall t \in \mathbf{T}} |C_{S_t}| \leq \sqrt{n}, S_t \text{ is the } t^{\text{th}} \text{ element of } \mathscr{S}\right)$ **then**
5: $\quad\quad$ Append i to $\mathbf{T}$
6: $\quad\quad u(x) = \sum_{\forall t \in \mathbf{T}} e_{S_t}(x)$
7: $\quad\quad$ **if** $\left(u(x) \text{ is non-degenerate}\right)$ **and** $\left(u(x) \text{ is orthogonal on each position (Lemma 12.1)}\right)$ **then**
8: $\quad\quad\quad U(z) = \text{MS}\,(u(x))$
9: $\quad\quad\quad k = n - \text{wt}_H\,(U(z))$
10: $\quad\quad\quad \mathscr{C} \Leftarrow$ a $[n, k, 1 + \text{wt}_H(u(x))]$ cyclic code defined by $u(x)$
11: $\quad\quad\quad$ **if** $\left(k \geq \frac{1}{4}\right)$ **and** $\left(\mathscr{C} \notin \textbf{CodeList}\right)$ **then**
12: $\quad\quad\quad\quad$ Add $\mathbf{C}$ to **CodeList**
13: $\quad\quad\quad$ **end if**
14: $\quad\quad$ **end if**
15: $\quad\quad$ CodeSearch($\mathbf{T}$, i)
16: $\quad$ **end if**
17: $\quad \mathbf{T} \Leftarrow \mathbf{T}_{\text{prev}}$
18: **end for**

Table 12.1 lists some example of codes obtained from Algorithm 12.1. Note that all codes with code rate less than 0.25 are excluded from the table and codes of longer lengths may also be constructed. We can also see that some of the codes in Table 12.1 have the same parameters as the Euclidean and projective geometry codes, which have been shown by Jin et al. [16] to perform well under iterative decoding.

Table 12.1 Examples of 2-cyclotomic coset-based LDPC codes

$[n, k, d]$	Cyclotomic cosets
[21, 11, 6]	C_7, C_9
[63, 37, 9]	C_{21}, C_{23}
[93, 47, 8]	C_3, C_{31}
[73, 45, 10]	C_1
[105, 53, 8]	C_7, C_{15}
[219, 101, 12]	C_3, C_{73}
[255, 135, 13]	C_1, C_{119}
[255, 175, 17]	C_1, C_{27}
[273, 191, 18]	C_1, C_{91}, C_{117}
[341, 205, 16]	C_1, C_{55}
[511, 199, 19]	C_5, C_{37}
[511, 259, 13]	C_1, C_{219}
[819, 435, 13]	C_1
[819, 447, 19]	C_1, C_{351}
[1023, 661, 23]	C_1, C_{53}, C_{341}
[1023, 781, 33]	$C_1, C_{53}, C_{123}, C_{341}$
[1057, 813, 34]	C_5, C_{43}, C_{151}
[1387, 783, 28]	C_1, C_{247}
[1971, 1105, 21]	C_1, C_{657}
[2047, 1167, 23]	C_1, C_{27}
[2325, 1335, 28]	C_1, C_{57}, C_{775}
[2325, 1373, 30]	C_1, C_{525}, C_{1085}
[2359, 1347, 22]	C_1
[3741, 2229, 29]	C_1
[3813, 2087, 28]	C_1, C_{369}, C_{1271}
[4095, 2767, 49]	$C_1, C_{41}, C_{235}, C_{733}$
[4095, 3367, 65]	$C_1, C_{41}, C_{235}, C_{273}, C_{411}, C_{733}$
[4161, 2827, 39]	C_1, C_{307}, C_{1387}
[4161, 3431, 66]	$C_1, C_{285}, C_{307}, C_{357}, C_{1387}$
[4681, 2681, 31]	C_1, C_{51}
[5461, 3781, 43]	C_1, C_{77}, C_{579}

A key feature of the cyclotomic coset-based construction is the ability to increment the minimum Hamming distance of a code by adding further weight from other idempotents and so steadily decrease the sparseness of the resulting parity-check matrix. Despite the construction method producing LDPC codes with no cycles of length 4, it is important to remark that codes that have cycles of length 4 in their parity-check matrices do not necessary have bad performance under iterative decoding, and a similar finding has been demonstrated by Tang et al. [31]. It has been observed

that there are many cyclotomic coset-based LDPC codes that have this property, and the constraints in Algorithm 12.1 can be easily relaxed to allow the construction of cyclic LDPC codes with girth 4.

12.2.1 Mattson–Solomon Domain Construction of Binary Cyclic LDPC Codes

The $[n, k, d]$ cyclic LDPC codes presented in Sect. 4.4 are constructed using the sum of idempotents, which are derived from the cyclotomic cosets modulo n, to define the parity-check matrix. A different insight into this construction technique may be obtained by working in the Mattson–Solomon domain.

Let n be a positive odd integer, $\mathbb{F}_{2^m}$ be a splitting field for $x^n - 1$ over $\mathbb{F}_2$, α be a generator for $\mathbb{F}_{2^m}$, and $T_m(x)$ be a polynomial with maximum degree of $n - 1$ and coefficients in $\mathbb{F}_{2^m}$. Similar to Sect. 4.4, the notation of $T(x)$ is used as an alternative to $T_1(x)$ and the variables x and z are used to distinguish the polynomials in the domain and codomain. Let the decomposition of $z^n - 1$ into irreducible polynomials over $\mathbb{F}_2$ be contained in a set $\mathscr{F} = \{f_1(z), f_2(z), \ldots, f_t(z)\}$, i.e. $\prod_{1 \le i \le t} f_i(z) = z^n - 1$. For each $f_i(z)$, there is a corresponding primitive idempotent, denoted as $\theta_i(z)$, which can be obtained by [20]

$$\theta_i(z) = \frac{z(z^n - 1) f_i'(z)}{f_i(z)} + \delta(z^n - 1) \tag{12.3}$$

where $f_i'(z) = \frac{d}{dz} f_i(z)$, $f_i'(z) \in T(z)$ and the integer δ is defined by

$$\delta = \begin{cases} 1 & \text{if } \deg(f_i(z)) \text{ is odd,} \\ 0 & \text{otherwise.} \end{cases}$$

Let the decomposition of $z^n - 1$ and its corresponding primitive idempotent be listed as follows:

$$\begin{matrix} u_1(x) & \theta_1(z) & f_1(z) \\ u_2(x) & \theta_2(z) & f_2(z) \\ \vdots & \vdots & \vdots \\ u_t(x) & \theta_t(z) & f_t(z). \end{matrix}$$

Here $u_1(x), u_2(x), \ldots, u_t(x)$ are the binary idempotents whose Mattson–Solomon polynomials are $\theta_1(z), \theta_2(z), \ldots, \theta_t(z)$, respectively. Assume that $\mathscr{I} \subseteq \{1, 2, \ldots, t\}$, let the binary polynomials $u(x) = \sum_{\forall i \in \mathscr{I}} u_i(x)$, $f(z) = \prod_{\forall i \in \mathscr{I}} f_i(z)$, and $\theta(z) =$

$\sum_{\forall i \in \mathscr{I}} \theta_i(z)$. It is apparent that, since $u_i(x) = \mathrm{MS}^{-1}(\theta_i(z))$, $u(x) = \mathrm{MS}^{-1}(\theta(z))$ and $u(x)$ is an idempotent.[1]

Recall that $u(x)$ is a low-weight binary idempotent whose reciprocal polynomial can be used to define the parity-check matrix of a cyclic LDPC code. The number of distinct n^{th} roots of unity which are also roots of the idempotent $u(x)$ determines the dimension of the resulting LDPC code. In this section, the design of cyclic LDPC codes is based on several important features of a code. These features, which are listed as follows, may be easily gleaned from the Mattson–Solomon polynomial of $u(x)$ and the binary irreducible factors of $z^n - 1$.

1. **Weight of the idempotent** $u(x)$
   The weight of $u(x)$ is the number of n^{th} roots of unity which are zeros of $f(z)$. Note that, $f(\alpha^i) = 0$ if and only if $\theta(\alpha^i) = 1$ since an idempotent takes only the values 0 and 1 over $\mathbb{F}_{2^m}$. If $u(x)$ is written as $u_0 + u_i x + \cdots + u_{n-1}x^{n-1}$, following (11.2), we have

$$u_i = \theta(\alpha^i) \pmod 2 \qquad \text{for } i = \{0, 1, \ldots, n-1\}.$$

   Therefore, $u_i = 1$ precisely when $f(\alpha^i) = 0$, giving $\mathrm{wt}_H(u(x))$ as the degree of the polynomial $f(z)$.
2. **Number of zeros of** $u(x)$
   Following (11.1), it is apparent that the number of zeros of $u(x)$ which are roots of unity, which is also the dimension of the code k, is

$$\text{Number of zeros of } u(x) = k = n - \mathrm{wt}_H(\theta(z)). \tag{12.4}$$

3. **Minimum Hamming distance bound**
   The lower bound of the minimum Hamming distance of a cyclic code, defined by idempotent $u(x)$, is given by its BCH bound, which is determined by the number of consecutive powers of α, taken cyclically (mod n), which are roots of the generating idempotent $e_g(x) = 1 + u(x)$. In the context of $u(x)$, it is the same as the number of consecutive powers of α, taken cyclically (mod n), which are not roots of $u(x)$. Therefore, it is the largest number of consecutive non-zero coefficients in $\theta(z)$, taken cyclically (mod n).

The method of finding $f_i(z)$ is well established and using the above information, a systematic search for idempotents of suitable weight may be developed. To be efficient, the search procedure has to start with an increasing order of $\mathrm{wt}_H(u(x))$ and this requires rearrangement of the set $\mathscr{F}$ such that $\deg(f_i(z)) < \deg(f_i i + 1(z))$ for all i. It is worth mentioning that it is not necessary to evaluate $u(x)$ by taking the

[1] Since the Mattson–Solomon polynomial of a binary polynomial is an idempotent and vice-versa [20], the Mattson–Solomon polynomial of a binary idempotent is also a binary idempotent.

Mattson–Solomon polynomial of $\theta(z)$, for each $f(z)$ obtained. It is more efficient to obtain $u(x)$ once the desired code criteria, listed above, are met.

For an exhaustive search, the complexity is of order $\mathcal{O}\left(2^{|\mathscr{F}|}\right)$. A search algorithm, see Algorithm 12.2, has been developed and it reduces the complexity considerably by targeting the search on the following key parameters. Note that this search algorithm, which is constructed in the Mattson–Solomon domain, is not constrained to find cyclic codes that have girth at least 6.

1. **Sparseness of the parity-check matrix**
   A necessary condition for the absence of cycles of length 4 is given by the inequality $\mathrm{wt}_H(u(x))\,(\mathrm{wt}_H(u(x)) - 1) \leq n - 1$. Since $\mathrm{wt}_H(u(x)) = \deg(f(z))$, a reasonable bound is

$$\sum_{\forall i \in \mathscr{I}} \deg(f_i(z)) \leq \sqrt{n}.$$

   In practise, this limit is extended to enable the finding of good cyclic LDPC codes which have girth of 4 in their underlying Tanner graph.
2. **Code rate**
   The code rate is directly proportional to the number of roots of $u(x)$. If R_{min} represents the minimum desired code rate, then it follows from (12.4) that we can refine the search to consider the cases where

$$\mathrm{wt}_H(\theta(z)) \leq (1 - R_{min})n\,.$$

3. **Minimum Hamming distance**
   If the idempotent $u(x)$ is orthogonal on each position, then the minimum Hamming distance of the resulting code defined by $u(x)$ is equal to $1 + \mathrm{wt}_H(u(x))$. However, for cyclic codes with cycles of length 4, there is no direct method to determine their minimum Hamming distance and the BCH bound provides a lower bound to the minimum Hamming distance. Let d be the lowest desired minimum Hamming distance and r_θ be the largest number of consecutive non-zero coefficients, taken cyclically, of $\theta(z)$. If a cyclic code has r_θ of d, then its minimum Hamming distance is at least $1 + d$. It follows that we can further refine the search with the constraint

$$r_\theta \geq d - 1.$$

In comparison to the construction method described in Sect. 4.4, we can see that the construction given in Sect. 4.4 starts from the idempotent $u(x)$, whereas this section starts from the idempotent $\theta(z)$, which is the Mattson–Solomon polynomial of $u(x)$. Both construction methods are equivalent and the same cyclic LDPC codes are produced.

Algorithm 12.2 MSCodeSearch($\mathbf{V}, index$)

Input:

$\mathbf{V} \Leftarrow$ a vector initialised to $\emptyset$
$index \Leftarrow$ an integer initialised to -1
$R_{min} \Leftarrow$ minimum code rate of interest
$d \Leftarrow$ lowest expected minimum distance
$\delta \Leftarrow$ small positive integer
$\mathbf{F}(z) \Leftarrow \{f_i(z)\}\ \forall i \in I$ sorted in ascending order of the degree
$\mathbf{Q}(z) \Leftarrow \{\theta_i(z)\}\ \forall i \in I$

Output:

CodesList contains set of codes

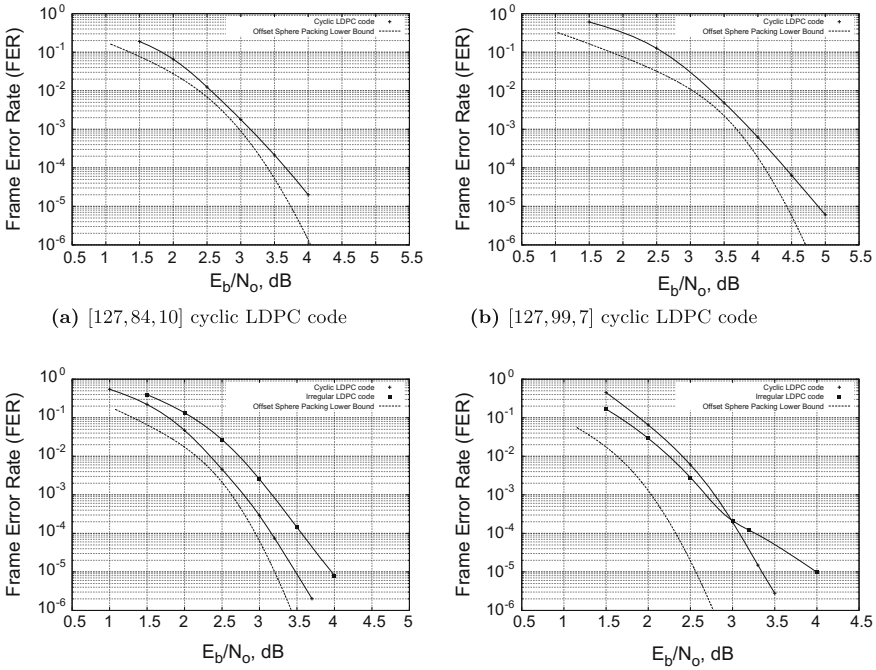
1: $\mathbf{T} \Leftarrow \mathbf{V}$
2: **for** $\left(i{=}index{+}1;\ i \leq |\mathscr{I}|;\ i{+}{+}\right)$ **do**
3: $\quad \mathbf{T}_{\text{prev}} \Leftarrow \mathbf{T}$
4: $\quad$ **if** $\left(\sum_{\forall j \in \mathbf{T}} \deg(f_j(x)) + \deg(f_i(x)) \leq \sqrt{n} + \delta\right)$ **then**
5: $\quad\quad$ Append i to $\mathbf{T}$
6: $\quad\quad \theta(z) \Leftarrow \sum_{\forall j \in \mathbf{T}} \theta_j(z)$
7: $\quad\quad$ **if** $\left(\text{wt}_H(\theta(z)) \leq (1 - R_{min})n$ **and** $r_\theta > d\right)$ **then**
8: $\quad\quad\quad u(x) \Leftarrow \text{MS}^{-1}(\theta(z))$
9: $\quad\quad\quad$ **if** $\left(u(x)$ is non-degenerate$\right)$ **then**
10: $\quad\quad\quad\quad \mathscr{C} \Leftarrow$ a cyclic code defined by $u(x)$
11: $\quad\quad\quad\quad$ **if** $\left(\mathscr{C} \notin$ **CodeList**$\right)$ **then**
12: $\quad\quad\quad\quad\quad$ Add **C** to **CodeList**
13: $\quad\quad\quad\quad$ **end if**
14: $\quad\quad\quad$ **end if**
15: $\quad\quad$ **end if**
16: $\quad\quad$ MSCodeSearch($\mathbf{T}, i$)
17: $\quad$ **end if**
18: $\quad \mathbf{T} \Leftarrow \mathbf{T}_{\text{prev}}$
19: **end for**

Some good cyclic LDPC codes with cycles of length 4 found using Algorithm 12.2, which may also be found using Algorithm 12.1, are tabulated in Table 12.2. A check based on Lemma 12.1 may be easily incorporated in Step 12 of Algorithm 12.2 to filter out cyclic codes whose Tanner graph has girth of 4.

Figure 12.3 demonstrates the FER performance of several cyclic LDPC codes found by Algorithm 12.2. It is assumed that binary antipodal signalling is employed and the iterative decoder uses the RVCM algorithm described by Papagiannis et al. [23]. The FER performance is compared against the sphere packing lower bound offset for binary transmission. We can see that the codes [127, 84, 10] and [127, 99, 7], despite having cycles of length 4, are around 0.3 dB from the offset sphere packing lower bound at 10^{-4} FER. Figure 12.3c compares two LDPC codes of block size 255 and dimension 175, an algebraic code obtained by Algorithm 12.2 and an irregular code constructed using the PEG algorithm [10]. It can be seen that, in addition to having improved minimum Hamming distance, the cyclic LDPC code is 0.4 dB superior to the irregular code, and compared to the offset sphere packing lower bound, it is within 0.25 dB away at 10^{-4} FER. The effect of the error floor is apparent in the FER performance of the [341, 205, 6] irregular LDPC code, as

Table 12.2 Several good cyclic LDPC codes with girth of 4

$[n, k, d]$	$u(x)$
[51, 26, 10]	$1+x^3+x^6+x^{12}+x^{17}+x^{24}+x^{27}+x^{34}+x^{39}+x^{45}+x^{48}$
[63, 44, 8]	$1+x^9+x^{11}+x^{18}+x^{21}+x^{22}+x^{25}+x^{27}+x^{36}+x^{37}+x^{42}+x^{44}+x^{45}+x^{50}+x^{54}$
[117, 72, 12]	$1+x+x^2+x^4+x^8+x^{11}+x^{16}+x^{22}+x^{32}+x^{44}+x^{59}+x^{64}+x^{88}$
[127, 84, 10]	$1+x+x^2+x^4+x^8+x^{16}+x^{32}+x^{55}+x^{59}+x^{64}+x^{91}+x^{93}+x^{109}+x^{110}+x^{118}$
[127, 91, 10]	$1+x^2+x^{10}+x^{18}+x^{29}+x^{32}+x^{33}+x^{49}+x^{50}+x^{54}+x^{58}+x^{65}+x^{74}+x^{76}+x^{78}+x^{86}+x^{87}+x^{88}+x^{92}+x^{93}+x^{95}$
[127, 92, 7]	$1+x^5+x^{10}+x^{20}+x^{29}+x^{31}+x^{33}+x^{39}+x^{40}+x^{58}+x^{62}+x^{66}+x^{78}+x^{79}+x^{80}+x^{83}+x^{103}+x^{105}+x^{115}+x^{116}+x^{121}+x^{124}$
[127, 99, 7]	$1+x^{13}+x^{16}+x^{18}+x^{22}+x^{26}+x^{39}+x^{42}+x^{45}+x^{46}+x^{49}+x^{57}+x^{65}+x^{68}+x^{70}+x^{78}+x^{80}+x^{90}+x^{91}+x^{92}+x^{96}+x^{97}+x^{102}+x^{103}+x^{105}+x^{108}+x^{111}$



(a) [127, 84, 10] cyclic LDPC code

(b) [127, 99, 7] cyclic LDPC code

(c) [255, 175, 17] cyclic and [255, 175, 6] irregular PEG LDPC codes

(d) [341, 205, 16] cyclic and [341, 205, 6] irregular PEG LDPC codes

Fig. 12.3 FER performance of some binary cyclic LDPC codes

shown in Fig. 12.3d. The floor of this irregular code is largely attributed to minimum Hamming distance error events. Whilst this irregular code, at low SNR region, has better convergence than does the algebraic LDPC code of the same block length and dimension, the benefit of having higher minimum Hamming distance is obvious as

the SNR increases. The [341, 205, 16] cyclic LDPC code is approximately 0.8 dB away from the offset sphere packing lower bound at 10^{-4} FER.

It is clear that short block length ($n \leq 350$) cyclic LDPC codes have outstanding performance and the gap to the offset sphere packing lower bound is relatively close. However, as the block length increases, the algebraic LDPC codes, although these code have large minimum Hamming distance, have a convergence issue, and the threshold to the waterfall region is at larger E_b/N_0. The convergence problem arises because as the minimum Hamming distance increases, the weight of the idempotent $u(x)$, which defines the parity-check matrix, also increases. In fact, if $u(x)$ satisfies Lemma 12.1, we know that $\mathrm{wt}_H(u(x)) = d - 1$, where d is the minimum Hamming distance of the code. Large values of $\mathrm{wt}_H(u(x))$ result in a parity-check matrix that is not as sparse as that of a good irregular LDPC code of the same block length and dimension.

12.2.2 Non-Binary Extension of the Cyclotomic Coset-Based LDPC Codes

The code construction technique for the cyclotomic coset-based binary cyclic LDPC codes, which is discussed in Sect. 4.4, may be extended to non-binary fields. Similar to the binary case, the non-binary construction produces the dual-code idempotent which is used to define the parity-check matrix of the associated LDPC code.

Let m and m' be positive integers with $m \mid m'$, so that $\mathbb{F}_{2^m}$ is a subfield of $\mathbb{F}_{2^{m'}}$. Let n be a positive odd integer and $\mathbb{F}_{2^{m'}}$ be the splitting field of $x^n - 1$ over $\mathbb{F}_{2^m}$, so that $n|2^{m'} - 1$. Let $r = (2^{m'} - 1)/n$, $l = (2^{m'} - 1)/(2^m - 1)$, α be a generator for $\mathbb{F}_{2^{m'}}$ and β be a generator of $\mathbb{F}_{2^m}$, where $\beta = \alpha^l$. Let $T_a(x)$ be the set of polynomials of degree at most $n - 1$ with coefficients in $\mathbb{F}_{2^a}$. For the case of $a = 1$, we may denote $T_1(x)$ by $T(x)$ for convenience.

The Mattson–Solomon polynomial and its corresponding inverse, (11.1) and (11.2), respectively, may be redefined as

$$A(z) = \mathrm{MS}\,(a(x)) = \sum_{j=0}^{n-1} a(\alpha^{-rj}) z^j \tag{12.5}$$

$$a(x) = \mathrm{MS}^{-1}\,(A(z)) = \frac{1}{n} \sum_{i=0}^{n-1} A(\alpha^{ri}) x^i \tag{12.6}$$

where $a(x) \in T_{m'}(x)$ and $A(z) \in T_{m'}(z)$.

Recall that a polynomial $e(x) \in T_m(x)$ is termed an idempotent if the property $e(x) = e(x)^2 \pmod{x^n - 1}$ is satisfied. Note that $e(x) \neq e(x^2) \pmod{x^n - 1}$ unless $m = 1$. The following definition shows how to construct an idempotent for binary and non-binary polynomials.

Definition 12.9 (*Cyclotomic Idempotent*) Assume that $\mathscr{N}$ be a set as defined in Sect. 4.4, let $s \in \mathscr{N}$ and let $C_{s,i}$ represent the $(i+1)$th element of C_s, the cyclotomic coset of $s \pmod{n}$. Assume that the polynomial $e_s(x) \in T_m(x)$ is given by

$$e_s(x) = \sum_{0 \le i \le |C_s|-1} e_{C_{s,i}} x^{C_{s,i}}, \tag{12.7}$$

where $|C_s|$ is the number of elements in C_s. In order for $e_s(x)$ to be an idempotent, its coefficients may be chosen in the following manner:

(i) if $m = 1$, $e_{C_{s,i}} = 1$,
(ii) otherwise, $e_{C_{s,i}}$ is defined recursively as follows:

$$\begin{aligned} &\text{for } i = 0,\ e_{C_{s,i}} \in \{1, \beta, \beta^2, \ldots, \beta^{2^m-2}\}, \\ &\text{for } i > 0,\ e_{C_{s,i}} = e^2_{C_{s,i-1}}. \end{aligned}$$

We refer to the idempotent $e_s(x)$ as a cyclotomic idempotent.

Definition 12.10 (*Parity-Check Idempotent*) Let $\mathscr{M} \subseteq \mathscr{N}$ and let $u(x) \in T_m(x)$ be

$$u(x) = \sum_{s \in \mathscr{M}} e_s(x). \tag{12.8}$$

The polynomial $u(x)$ is an idempotent and it is called a parity-check idempotent.

As in Sect. 4.4, the parity-check idempotent $u(x)$ is used to define the $\mathbb{F}_{2^m}$ cyclic LDPC code over $\mathbb{F}_{2^m}$, which may be denoted by $[n, k, d]_{2^m}$. The parity-check matrix consists of n cyclic shifts of $x^{\deg(u(x))}u(x^{-1})$. For the non-binary case, the minimum Hamming distance d of the cyclic code is bounded by

$$d_0 + 1 \le d \le \min\left(\mathrm{wt}_H(g(x)), \mathrm{wt}_H(1 + u(x))\right),$$

where d_0 is the maximum run of consecutive ones in $U(z) = \mathrm{MS}(u(x))$, taken cyclically mod n.

Based on the description given above, a procedure to construct a cyclic LDPC code over $\mathbb{F}_{2^m}$ is as follows.

1. For integers m and n, obtain the splitting field $(\mathbb{F}_{2^{m'}})$ of $x^n - 1$ over $\mathbb{F}_{2^m}$. Unless the condition of $m \mid m'$ is satisfied, $\mathbb{F}_{2^m}$ cyclic LDPC code of length n cannot be constructed.
2. Generate the cyclotomic cosets modulo $2^{m'} - 1$ denoted as C'.
3. Derive a polynomial $p(x)$ from C' and let $s \in \mathscr{N}$ be the smallest positive integer such that $|C'_s| = m$. The polynomial $p(x)$ is the minimal polynomial of α^s,

$$p(x) = \prod_{0 \le i < m} \left(x + \alpha^{C'_{s,i}}\right). \tag{12.9}$$

Construct all elements of $\mathbb{F}_{2^m}$ using $p(x)$ as the primitive polynomial.
4. Let C be the cyclotomic cosets modulo n and let $\mathscr{N}$ be a set containing the smallest number in each coset of C. Assume that there exists a non-empty set $\mathscr{M} \subset \mathscr{N}$ and following Definition 12.10 construct the parity-check idempotent $u(x)$. The coefficients of $u(x)$ can be assigned following Definition 12.9.
5. Generate the parity-check matrix of $\mathscr{C}$ using the n cyclic shifts of $x^{\deg(u(x))}u(x^{-1})$.
6. Compute r and l, and then take the Mattson–Solomon polynomial of $u(x)$ to produce $U(z)$. Obtain the code dimension and the lower bound of the minimum Hamming distance from $U(z)$.

Example 12.3 Consider the construction of a $n = 21$ cyclic LDPC code over $\mathbb{F}_{2^6}$. The splitting field of $x^{21} - 1$ over $\mathbb{F}_{2^6}$ is $\mathbb{F}_{2^6}$, and this implies that $m = m' = 6$, $r = 3$ and $l = 1$. Let C and C' denote the cyclotomic cosets modulo n and $2^{m'} - 1$, respectively. We know that $|C'_1| = 6$ and therefore the primitive polynomial $p(x)$ has roots of α^j, for all $j \in C'_1$, i.e. $p(x) = 1 + x + x^6$. By letting $1 + \beta + \beta^6 = 0$, all of the elements of $\mathbb{F}_{2^6}$ can be defined. If $u(x)$ is the parity-check idempotent generated by the sum of the cyclotomic idempotents defined by C_s, where $s \in \{\mathscr{M} : 5, 7, 9\}$ and $e_{C_{s,0}}$ for all $s \in \mathscr{M}$ be β^{23}, 1 and 1, respectively,

$$\begin{aligned} u(x) = {} & \beta^{23}x^5 + x^7 + x^9 + \beta^{46}x^{10} + \beta^{43}x^{13} + x^{14} + x^{15} + \beta^{53}x^{17} + x^{18} \\ & + \beta^{58}x^{19} + \beta^{29}x^{20} \end{aligned}$$

and its Mattson–Solomon polynomial $U(z)$ indicates that it is a $[21, 15, \geq 5]_{2^6}$ cyclic code, whose binary image is a [126, 90, 8] linear code.

The following systematic search algorithm is based on summing each possible combination of the cyclotomic idempotents to search for all possible $\mathbb{F}_{2^m}$ cyclic codes of a given length. As in Algorithm 12.2, the search algorithm targets the following key parameters:

1. **Sparseness of the resulting parity-check matrix**
   Since the parity-check matrix is directly derived from $u(x)$ which consists of the sum of the cyclotomic idempotents, only low-weight cyclotomic idempotents are of interest. Let W_{max} be the maximum $\mathrm{wt}_H(u(x))$; then the search algorithm will only choose the cyclotomic idempotents whose sum has total weight less than or equal to W_{max}.
2. **High code rate**
   The number of roots of $u(x)$ which are also roots of unity define the dimension of the resulting LDPC code. Let the integer k_{min} be defined as the minimum code dimension, and the cyclotomic idempotents that are of interest are those whose Mattson–Solomon polynomial has at least k_{min} zeros.
3. **High minimum Hamming distance**
   Let the integer d' be the smallest value of the minimum Hamming distance of the code. The sum of the cyclotomic idempotents should have at least $d' - 1$ consecutive powers of β which are roots of unity but not roots of $u(x)$.

Table 12.3 Examples of $[n, k, d]_{2^m}$ cyclic LDPC codes

q	$[n, k]$	$u(x)$	d	$d_b^{\dagger}$	Comment
$\mathbb{F}_{2^2}$	[51, 29]	$\beta^2x^3+\beta x^6+\beta^2x^{12}+\beta^2x^{17}+\beta x^{24}+\beta x^{27}+\beta x^{34}+\beta^2x^{39}+\beta x^{45}+\beta^2x^{48}$	$\geq$5	10	$m=2$, $m'=8$, $r=5$ and $l=85$
	[255, 175]	$\beta x^7+\beta^2x^{14}+\beta x^{28}+\beta^2x^{56}+x^{111}+\beta x^{112}+x^{123}+\beta^2x^{131}+x^{183}+x^{189}+\beta x^{193}+x^{219}+x^{222}+\beta^2x^{224}+x^{237}+x^{246}$	$\geq$17	$\leq$20	$m=2$, $m'=8$, $r=1$ and $l=85$
	[273, 191]	$\beta^2x^{23}+\beta x^{37}+\beta x^{46}+\beta^2x^{74}+\beta x^{91}+\beta^2x^{92}+\beta^2x^{95}+\beta^2x^{107}+x^{117}+\beta x^{148}+\beta^2x^{155}+\beta^2x^{182}+\beta x^{184}+\beta x^{190}+x^{195}+\beta x^{214}+x^{234}$	$\geq$18	$\leq$20	$m=2$, $m'=12$, $r=15$ and $l=1365$
$\mathbb{F}_{2^3}$	[63, 40]	$1+\beta^5x^9+\beta x^{13}+\beta^3x^{18}+\beta^2x^{19}+\beta^2x^{26}+\beta^6x^{36}+\beta^4x^{38}+\beta x^{41}+\beta^4x^{52}$	$\geq$6	10	$m=3$, $m'=6$, $r=1$ and $l=9$
	[63, 43]	$\beta^2x^9+\beta^3x^{11}+\beta^4x^{18}+x^{21}+\beta^6x^{22}+\beta^3x^{25}+x^{27}+\beta x^{36}+\beta^5x^{37}+x^{42}+\beta^5x^{44}+x^{45}+\beta^6x^{50}+x^{54}$	$\geq$8	$\leq$12	$m=3$, $m'=6$, $r=1$ and $l=9$
	[91, 63]	$\beta^6x+\beta^5x^2+\beta^3x^4+\beta^6x^8+\beta x^{13}+\beta^5x^{16}+\beta^5x^{23}+\beta^2x^{26}+\beta^3x^{32}+\beta^5x^{37}+\beta^3x^{46}+\beta^4x^{52}+\beta^6x^{57}+\beta^6x^{64}+\beta^3x^{74}$	$\geq$8	$\leq$10	$m=3$, $m'=12$, $r=45$ and $l=585$
$\mathbb{F}_{2^4}$	[85, 48]	$1+\beta^{12}x^{21}+\beta^9x^{42}+\beta^6x^{53}+\beta^3x^{69}+\beta^9x^{77}+\beta^{12}x^{81}+\beta^6x^{83}+\beta^3x^{84}$	$\geq$7	$\leq$12	$m=4$, $m'=8$, $r=3$ and $l=17$
$\mathbb{F}_{2^5}$	[31, 20]	$1+\beta^{28}x^5+\beta^7x^9+\beta^{25}x^{10}+x^{11}+x^{13}+\beta^{14}x^{18}+\beta^{19}x^{20}+x^{21}+x^{22}+x^{26}$	$\geq$7	12	$m=5$, $m'=5$, $r=l$ and $l=1$
	[31, 21]	$\beta^{23}x^5+\beta^{29}x^9+\beta^{15}x^{10}+\beta x^{11}+\beta^4x^{13}+\beta^{27}x^{18}+\beta^{30}x^{20}+\beta^{16}x^{21}+\beta^2x^{22}+\beta^8x^{26}$	$\geq$4	8	$m=5$, $m'=5$, $r=1$ and $l=1$
$\mathbb{F}_{2^6}$	[21, 15]	$\beta^{23}x^5+x^7+x^9+\beta^{46}x^{10}+\beta^{43}x^{13}+x^{14}+x^{15}+\beta^{53}x^{17}+x^{18}+\beta^{58}x^{19}+\beta^{29}x^{20}$	$\geq$5	8	$m=6$, $m'=6$, $r=3$ and $l=1$

†The minimum Hamming distance of the binary image which has been determined using the improved Zimmermann algorithm, Algorithm 5.1

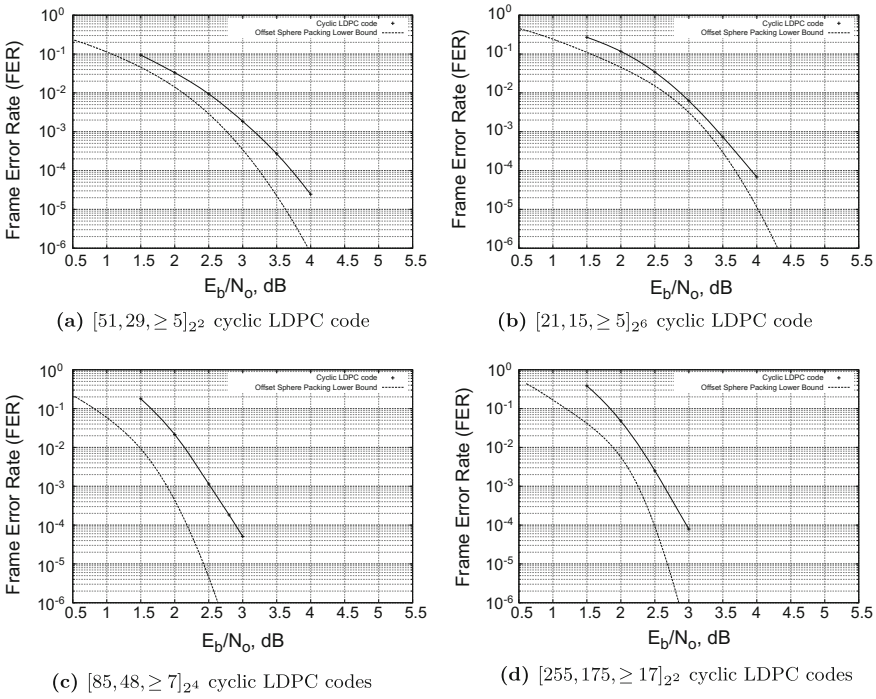
Following Definition 12.10 and the Mattson–Solomon polynomial

$$U(z) = \mathrm{MS}\left(\sum_{s\in\mathscr{M}} e_s(x)\right) = \sum_{s\in\mathscr{M}} E_s(z),$$

it is possible to maximise the run of the consecutive ones in $U(z)$ by varying the coefficients of $e_s(x)$. It is therefore important that all possible non-zero values of $e_{C_{s,0}}$ for all $s \in \mathscr{M}$ are included to guarantee that codes with the highest possible minimum Hamming distance are found.

Table 12.3 outlines some examples of $[n, k, d]_{2^m}$ cyclic LDPC codes. The non-binary algebraic LDPC codes in this table perform well under iterative decoding as shown in Fig. 12.4 assuming binary antipodal signalling and the AWGN channel. The RVCM algorithm is employed in the iterative decoder. The FER performance of these non-binary codes is compared to the offset sphere packing lower bound in Fig. 12.4.

As mentioned in Sect. 12.1.2, there is an inverse relationship between the convergence of the iterative decoder and the minimum Hamming distance of a code. The algebraic LDPC codes, which have higher minimum Hamming distances compared to irregular LDPC codes, do not converge well at long block lengths. It appears that



(a) $[51, 29, \geq 5]_{2^2}$ cyclic LDPC code

(b) $[21, 15, \geq 5]_{2^6}$ cyclic LDPC code

(c) $[85, 48, \geq 7]_{2^4}$ cyclic LDPC codes

(d) $[255, 175, \geq 17]_{2^2}$ cyclic LDPC codes

Fig. 12.4 FER performance of some non-binary cyclic LDPC codes

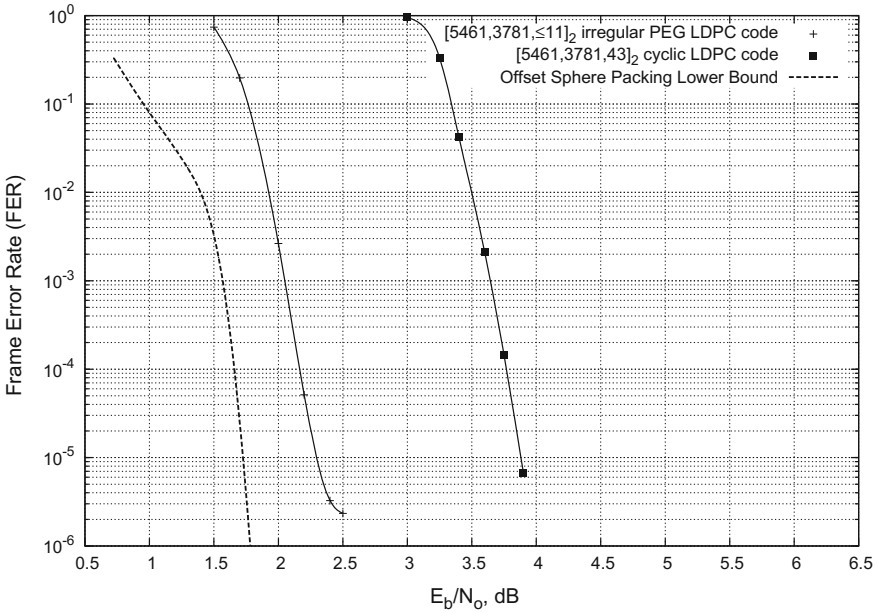


Fig. 12.5 FER performance of algebraic and irregular LDPC codes of rate 0.6924 and code length 5461 bits

the best convergence at long code lengths can only be realised by irregular LDPC codes with good degree distributions. Figure 12.5 shows the performance of two LDPC codes of block length 5461 bits and code rate 0.6924; one is an irregular code constructed using the PEG algorithm and the other one is an algebraic code of minimum Hamming distance 43 based on cyclotomic coset and idempotent construction (see Table 12.1). These results are for the AWGN channel using binary antipodal signalling with a belief propagation iterative decoder featuring 100 iterations. We can see that at 10^{-5} FER, the irregular PEG code is superior by approximately 1.6 dB compared to the algebraic cyclic LDPC code. However, for short code lengths, algebraic LDPC codes are superior. The codes have better performance and have simpler encoders than ad hoc designed LDPC codes.

12.3 Irregular LDPC Codes from Progressive Edge-Growth Construction

It is shown by Hu et al. [11] that LDPC codes obtained using the PEG construction method can perform better than other types of randomly constructed LDPC codes. The PEG algorithm adds edges to each vertex such that the local girth is maximised. The PEG algorithm considers only the variable degree sequence, and the check degree

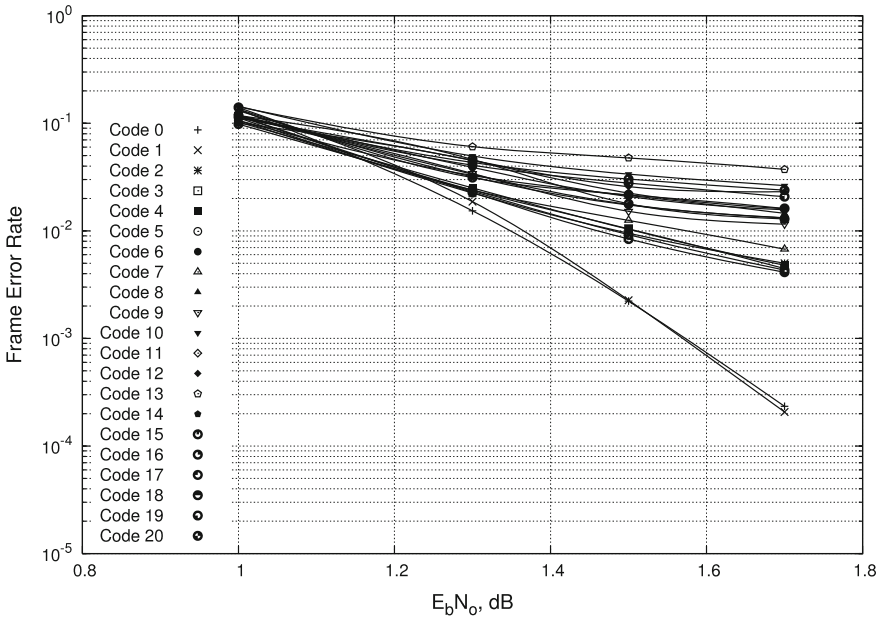


Fig. 12.6 Effect of vertex degree ordering in PEG algorithm

sequence is maintained to be as uniform as possible. In this section, the results of experimental constructions of irregular LDPC codes using the PEG algorithm are presented. Analysis on the effects of the vertex degree ordering and degree sequences have been carried out by means of computer simulations. All simulation results in this section, unless otherwise stated, were obtained using binary antipodal signalling with the belief propagation decoder using 100 iterations. Each simulation run was terminated after the decoder had produced 100 erroneous frames.

Figure 12.6 shows the FER performance of various [2048, 1024] irregular LDPC codes constructed using the PEG algorithm with different vertex degree orderings. These LDPC codes have variable degree sequence $\Lambda_\lambda(x) = 0.475x^2 + 0.280x^3 + 0.035x^4 + 0.109x^5 + 0.101x^{15}$. Let $(v_0, v_1, \ldots, v_i, \ldots, v_{n-1})$ be a set of variable vertices of an LDPC code. Code 0 and Code 1 LDPC codes were constructed with an increasing vertex degree ordering, i.e. $\deg(v_0) \leq \deg(v_1) \leq \cdots \leq \deg(v_{n-1})$, whereas the remaining LDPC codes were constructed with random vertex degree ordering.

Figure 12.6 clearly shows that, unless the degree of the variable vertices is assigned in an increasing order, poor LDPC codes are obtained. In random degree ordering of half rate codes, it is very likely to encounter the situation where, as the construction approaches the end, there are some low-degree variable vertices that have no edge connected to them. Since almost all of the variable vertices would have already had edges connected to them, the low-degree variable vertices would not have many choice of edges to connect in order to maximise the local girth. It has been observed

that, in many cases, these low-degree variable vertices are connected to each other, forming a cycle which involves all vertices, and the resulting LDPC codes often have a low minimum Hamming distance. If d variable vertices are connected to each other and a cycle of length $2d$ is formed, then the minimum Hamming distance of the resulting code is d because the sum of these d columns in the corresponding parity-check matrix $\boldsymbol{H}$ is $\mathbf{0}^T$.

In contrast, for the alternative construction which starts from an increasing degree of the variable vertices, edges are connected to the low-degree variable vertices earlier in the process. Short cycles, which involve the low-degree variable vertices and lead to low minimum Hamming distance, may be avoided by ensuring these low-degree variable vertices have edges connected to the parity-check vertices which are connected to high-degree variable vertices.

It can be expected that the PEG algorithm will almost certainly produce poor LDPC codes if the degree of the variable vertices is assigned in descending order. It is concluded that all PEG-based LDPC codes should be constructed with increasing variable vertex degree ordering.

Figure 12.7 shows the effect of low-degree variable vertices, especially λ_2 and λ_3, on the FER performance of various [512, 256] PEG-constructed irregular LDPC codes. Table 12.4 shows the variable degree sequences of the simulated irregular codes. Figure 12.7 indicates that, with the fraction of high-degree variable vertices kept constant, the low-degree variable vertices have influence over the convergence

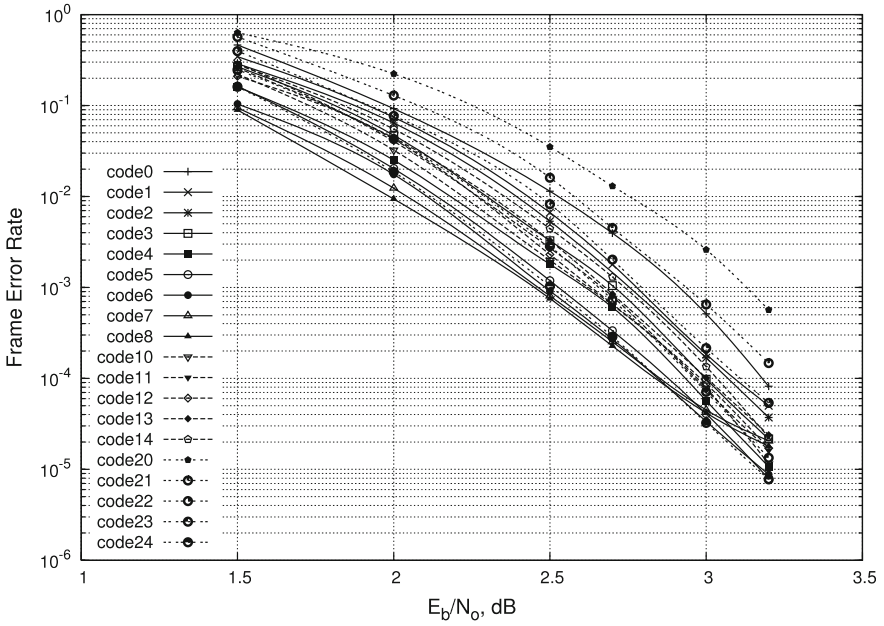


Fig. 12.7 Effect of low-degree variable vertices

Table 12.4 Variable degree sequences for codes in Fig. 12.7

Code	λ_2	λ_3	λ_4	λ_5	λ_{14}
Code 0	0.150	0.350	0.350	0.050	0.100
Code 1	0.200	0.325	0.325	0.050	0.100
Code 2	0.250	0.300	0.300	0.050	0.100
Code 3	0.300	0.275	0.275	0.050	0.100
Code 4	0.350	0.250	0.250	0.050	0.100
Code 5	0.400	0.225	0.225	0.050	0.100
Code 6	0.450	0.200	0.200	0.050	0.100
Code 7	0.500	0.175	0.175	0.050	0.100
Code 8	0.550	0.150	0.150	0.050	0.100
Code 10	0.150	0.700	0.000	0.050	0.100
Code 11	0.200	0.550	0.100	0.050	0.100
Code 12	0.250	0.400	0.200	0.050	0.100
Code 13	0.300	0.250	0.300	0.050	0.100
Code 14	0.350	0.100	0.400	0.050	0.100
Code 20	0.150	0.000	0.700	0.050	0.100
Code 21	0.200	0.100	0.550	0.050	0.100
Code 22	0.250	0.200	0.400	0.050	0.100
Code 23	0.300	0.300	0.250	0.050	0.100
Code 24	0.350	0.400	0.100	0.050	0.100

in the waterfall region. As the fraction of low-degree variable vertices is increased, the FER in the low signal-to-noise ratio (SNR) region improves. On the other hand, LDPC codes with a high fraction of low-degree variable vertices tend to have low minimum Hamming distance and as expected, these codes exhibit early error floors. This effect is clearly depicted by Code 7 and Code 8, which have the highest fraction of low-degree variable vertices among all the codes in Fig. 12.7. Of all of the codes, Code 6 and Code 24 appear to have the best performance.

Figure 12.8 demonstrates the effect of high-degree variable vertices on the FER performance. These codes are rate 3/4 irregular LDPC codes of length 1024 bits with the same degree sequences, apart from their maximum variable vertex degree. One group has maximum degree of 8 and the other group has maximum degree of 12. From Fig. 12.8, it is clear that the LDPC codes with maximum variable vertex degree of 12 converge better under iterative decoding than those codes with maximum variable vertex degree of 8.

In a similar manner to Fig. 12.7, the effect of having various low-degree variable vertices is also demonstrated in Fig. 12.9. In this case, the LDPC codes are constructed to have the advantageous linear-time encoding complexity, where the parity symbols are commonly described as having a zigzag pattern [26]. In this case, λ_1 and λ_2 of these LDPC codes are fixed and the effect of varying λ_3, λ_4 and λ_5 is investigated.

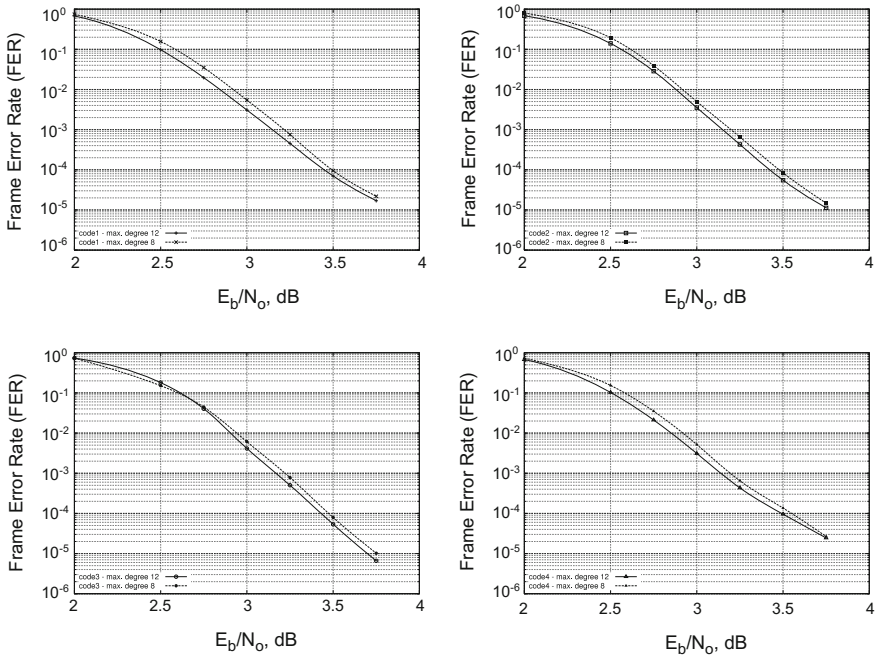


Fig. 12.8 Effect of high-degree variable vertices

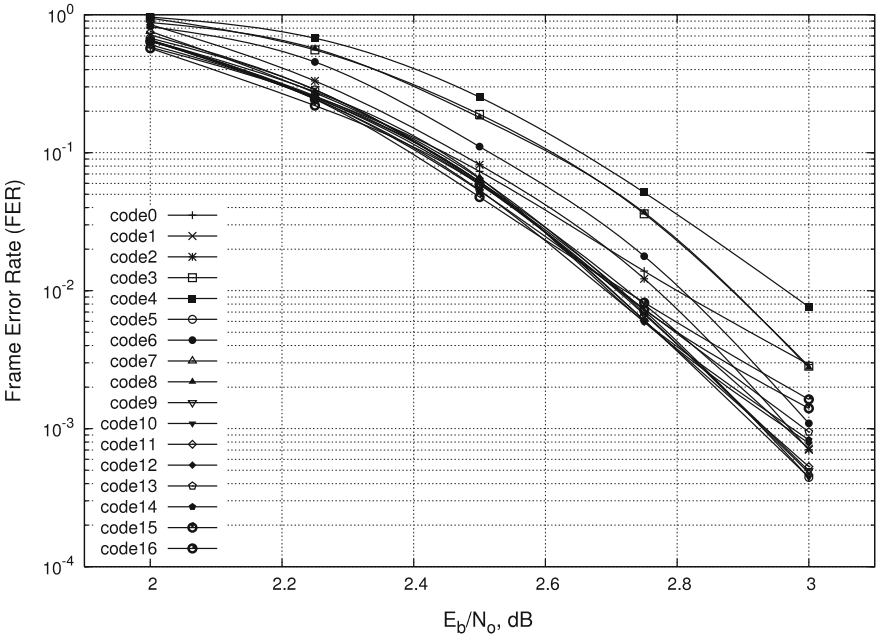


Fig. 12.9 Effect of varying low-degree variable vertices

Table 12.5 Variable degree sequences of LDPC codes in Fig. 12.9

Code	λ_1	λ_2	λ_3	λ_4	λ_5	λ_{12}
Code 0	0.000625	0.249375	0.644375			0.105625
Code 1	0.000625	0.249375	0.420000	0.224375		0.105625
Code 2	0.000625	0.249375	0.195000	0.449375		0.105625
Code 3	0.000625	0.249375		0.420000	0.224375	0.105625
Code 4	0.000625	0.249375		0.195000	0.449375	0.105625
Code 5	0.000625	0.249375	0.420000	0.111875	0.111875	0.106250
Code 6	0.000625	0.249375	0.195000	0.224375	0.224375	0.106250
Code 7	0.000625	0.249375	0.420000		0.224375	0.105625
Code 8	0.000625	0.249375	0.195000		0.449375	0.105625
Code 9	0.000625	0.249375	0.449375		0.195000	0.105625
Code 10	0.000625	0.249375	0.449375	0.097500	0.097500	0.105625
Code 11	0.000625	0.249375	0.449375	0.044375	0.150000	0.106250
Code 12	0.000625	0.249375	0.495000		0.150000	0.105000
Code 13	0.000625	0.249375	0.495000	0.075000	0.075000	0.105000
Code 14	0.000625	0.249375	0.495000	0.037500	0.111875	0.105625
Code 15	0.000625	0.249375	0.570000		0.075000	0.105000
Code 16	0.000625	0.249375	0.570000	0.037500	0.037500	0.105000

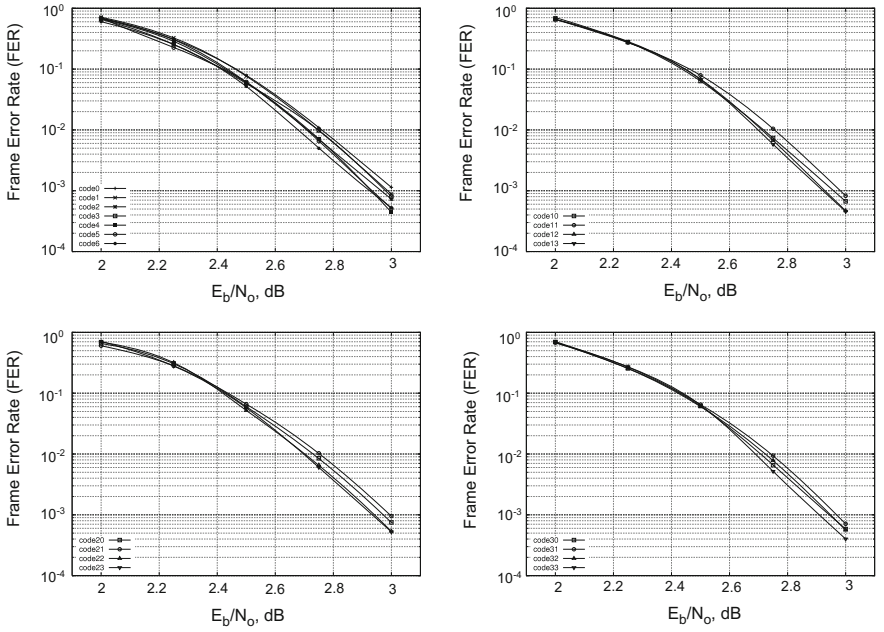


Fig. 12.10 Effect of varying high-degree variable vertices

Table 12.6 Variable degree sequences of LDPC codes in Fig. 12.10

Code	λ_3	λ_4	λ_5	λ_8	λ_9	λ_{10}	λ_{11}	λ_{12}	λ_{13}	λ_{14}
Code 0	0.420000	0.111875	0.111875	0.10625						
Code 1	0.420000	0.111875	0.111875		0.10625					
Code 2	0.420000	0.111875	0.111875			0.10625				
Code 3	0.420000	0.111875	0.111875				0.10625			
Code 4	0.420000	0.111875	0.111875					0.10625		
Code 5	0.420000	0.111875	0.111875						0.10625	
Code 6	0.420000	0.111875	0.111875							0.10625
Code 10	0.434375	0.111875	0.111875					0.091875		
Code 11	0.449375	0.111875	0.111875					0.076875		
Code 12	0.405000	0.111875	0.111875					0.121250		
Code 13	0.390000	0.111875	0.111875					0.136250		
Code 20	0.420000	0.127500	0.111875					0.090625		
Code 21	0.420000	0.142500	0.111875					0.075625		
Code 22	0.420000	0.097500	0.111875					0.120625		
Code 23	0.420000	0.082500	0.111875					0.135625		
Code 30	0.420000	0.111875	0.127500					0.090625		
Code 31	0.420000	0.111875	0.142500					0.075625		
Code 32	0.420000	0.111875	0.097500					0.120625		
Code 33	0.420000	0.111875	0.082500					0.135625		

Note that $\lambda_1 = 0.000625$ and $\lambda_2 = 0.249375$

The variable degree sequences of the LDPC codes under investigation, which are rate 3/4 codes of length 1600 bits, are depicted in Table 12.5. The results show that, as in the previous cases, these low-degree variable vertices contribute to the waterfall region of the FER curve. The contribution of λ_i is more significant than that of λ_{i+1} and this may be observed by comparing the FER curves of Code 1 with either Code 3 or Code 4, which has λ_3 of 0.0. We can also see that Code 0, which has the most variable vertices of low degree, exhibits a high error floor.

In contrast to Fig. 12.9, Fig. 12.10 shows the effect of varying high-degree variable vertices. The LDPC codes considered here all have the same code rate and code length as those in Fig. 12.9 and their variable degree sequences are shown in Table 12.6. The results show that

- The contribution of the high-degree variable vertices is in the high SNR region. Consider Code 10 to Code 33, those LDPC codes that have larger λ_{12} tend to be more resilient to errors in the high SNR region than those with smaller λ_{12}. At $E_b/N_o = 3.0$ dB, Code 10, Code 11 and Code 12 are inferior to Code 13 and similarly, Code 23 and Code 33 have the best performance in their group.
- Large values of maximum variable vertex degree may not always lead to improved FER performance. For example, Code 5 and Code 6 do not perform as well as Code 4 at $E_b/N_o = 3.0$ dB. This may be explained as follows. As the maximum variable vertex degree is increased, some of the variable vertices have many edges connected to them, in the other words the corresponding symbols are checked by many parity-check equations. This increases the chances of having unreliable information from some of these equations during iterative decoding. In addition, a larger maximum variable vertex degree also increases the number of short cycles in the underlying Tanner graph of the code. It was concluded also by McEliece et al. [22] and by Etzion et al. [7] that short cycles lead to negative contributions preventing the convergence of the iterative decoder.

12.4 Quasi-cyclic LDPC Codes and Protographs

Despite irregular LDPC codes having lower error rates than their regular counterparts, Luby et al. [18], the extra complexity of the encoder and decoder hardware structure, has made this class of LDPC codes unattractive from an industry point of view. In order to encode an irregular code which has a parity-check matrix $\boldsymbol{H}$, Gaussian elimination has to be done to transform this matrix into reduced echelon form. Irregular LDPC codes, as shown in Sect. 12.3, may also be constructed by constraining the $n - k$ low-degree variable vertices of the Tanner graph to form a zigzag pattern, as pointed out by Ping et al. [26]. Translating these $n - k$ variable vertices of the Tanner graph into matrix form, we have

$$\boldsymbol{H}_p = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & \ddots & \ddots & \\ & & 1 & 1 \\ & & & 1 \end{bmatrix}. \tag{12.10}$$

The matrix $\boldsymbol{H}_p$ is non-singular and the columns of this matrix may be used as the coordinates of the parity-check bits of an LDPC code.

The use of zigzag parity checks does simplify the derivation of the encoder as the Gaussian elimination process is no longer necessary and encoding, assuming that

$$\boldsymbol{H} = \left[\boldsymbol{H}_u | \boldsymbol{H}_p\right]$$

$$= \begin{array}{c|cccc|ccccccc} v_0 & v_1 & \cdots & v_{k-2} & v_{k-1} & v_k & v_{k+1} & \cdots & v_{n-2} & v_{n-1} \\ \hline u_{0,0} & u_{0,1} & \cdots & u_{0,k-2} & u_{0,k-1} & 1 & & & & \\ u_{1,0} & u_{1,1} & \cdots & u_{1,k-2} & u_{1,k-1} & 1 & 1 & & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \ddots & \ddots & & \\ u_{n-k-2,0} & u_{n-k-2,1} & \cdots & u_{n-k-2,k-2} & u_{n-k-2,k-1} & & & 1 & 1 & \\ u_{n-k-1,0} & u_{n-k-1,1} & \cdots & u_{n-k-1,k-2} & u_{n-k-1,k-1} & & & & 1 & 1 \end{array},$$

can be performed by calculating parity-check bits as follows:

$$v_k = \sum_{j=0}^{k-1} v_j u_{0,j} \pmod 2$$

$$v_i = v_{i-1} + \sum_{j=0}^{k-1} v_j u_{i-k,j} \pmod 2 \qquad \text{for } k+1 \le i \le n-1\,.$$

Nevertheless, zigzag parity bit checks do not lead to a significant reduction in encoder storage space as the matrix $\boldsymbol{H}_u$ still needs to be stored. It is necessary to introduce additional structure in $\boldsymbol{H}_u$, such as using a quasi-cyclic property, to reduce significantly the storage requirements of the encoder.

12.4.1 Quasi-cyclic LDPC Codes

Quasi-cyclic codes have the property that each codeword is a m-sized cyclic shift of another codeword, where m is an integer. With this property simple feedback shift registers may be used for the encoder. This type of code is known as circulant codes defined by circulant polynomials and depending on the polynomials can have significant mathematical structure as described in Chap. 9. A circulant matrix is a square matrix where each row is a cyclic shift of the previous row and the first row

is the cyclic shift of the last row. In addition, each column is also a cyclic shift of the previous column and the column weight is equal to the row weight.

A circulant matrix is defined by a polynomial $r(x)$. If $r(x)$ has degree $<m$, the corresponding circulant matrix is an $m \times m$ square matrix. Let $\boldsymbol{R}$ be a circulant matrix defined by $r(x)$, then $\boldsymbol{M}$ is of the form

$$\boldsymbol{R} = \begin{bmatrix} r(x) \pmod{x^m - 1} \\ xr(x) \pmod{x^m - 1} \\ \vdots \\ x^i r(x) \pmod{x^m - 1} \\ \vdots \\ x^{m-1} r(x) \pmod{x^m - 1} \end{bmatrix} \tag{12.11}$$

where the polynomial in each row can be represented by an m-dimensional vector, which contains the coefficients of the corresponding polynomial. A quasi-cyclic code can be built from the concatenation of circulant matrices to define the generator or parity-check matrix.

Example 12.4 A quasi-cyclic code with defining polynomials $r_1(x) = 1 + x + x^3$ and $r_2(x) = 1 + x^2 + x^5$, where both polynomials have degree less than the maximum degree of 6, produces a parity-check matrix in the following form:

$$\boldsymbol{H} = \left[\begin{array}{ccccccc|ccccccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{array}\right].$$

Definition 12.11 (*Permutation Matrix*) A permutation matrix is a type of circulant matrix where each row or column has weight of 1. A permutation matrix, which is denoted by $\boldsymbol{P}_{m,j}$, has $r(x) = x^j \pmod{x^m - 1}$ as the defining polynomial and it satisfies the property that $\boldsymbol{P}_{m,j}^2 = \boldsymbol{I}_m$, where $\boldsymbol{I}_m$ is an $m \times m$ identity matrix.

Due to the sparseness of the permutation matrix, these are usually used to construct quasi-cyclic LDPC codes. The resulting LDPC codes produce a parity-check matrix in the following form:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{P}_{m,O_{0,0}} & \boldsymbol{P}_{m,O_{0,1}} & \cdots & \boldsymbol{P}_{m,O_{0,t-1}} \\ \boldsymbol{P}_{m,O_{1,0}} & \boldsymbol{P}_{m,O_{1,1}} & \cdots & \boldsymbol{P}_{m,O_{1,t-1}} \\ \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{P}_{m,O_{s-1,0}} & \boldsymbol{P}_{m,O_{s-1,1}} & \cdots & \boldsymbol{P}_{m,O_{s-1,t-1}} \end{bmatrix} \tag{12.12}$$

From (12.12), we can see that there exists a $s \times t$ matrix, denoted by $\boldsymbol{O}$, in $\boldsymbol{H}$. This matrix is called an *offset matrix* and it represents the exponent of $r(x)$ in each permutation matrix, i.e.

$$\boldsymbol{O} = \begin{bmatrix} O_{0,0} & O_{0,1} & \dots & O_{0,t-1} \\ O_{1,0} & O_{1,1} & \dots & O_{1,t-1} \\ \vdots & \vdots & \vdots & \vdots \\ O_{s-1,0} & O_{s-1,1} & \dots & O_{s-1,t-1} \end{bmatrix}$$

where $0 \leq O_{i,j} \leq m-1$, for $0 \leq i \leq s-1$ and $0 \leq j \leq t-1$. The permutation matrix $\boldsymbol{P}_{m,j}$ has m rows and m columns, and since the matrix $\boldsymbol{H}$ contains s and t of these matrices per row and column, respectively, the resulting code is a $[mt, m(t-s), d]$ quasi-cyclic LDPC code over $\mathbb{F}_2$.

In general, some of the permutation matrices $\boldsymbol{P}_{i,j}$ in (12.12) may be zero matrices. In this case, the resulting quasi-cyclic LDPC code is irregular and $O_{i,j}$ for which $\boldsymbol{P}_{i,j} = \boldsymbol{O}$ may be ignored. If none of the permutation matrices in (12.12) is a zero matrix, the quasi-cyclic LDPC code defined by (12.12) is a (s, t) regular LDPC code.

12.4.2 Construction of Quasi-cyclic Codes Using a Protograph

A protograph is a miniature prototype Tanner graph of arbitrary size, which can be used to construct a larger Tanner graph by means of replicate and permute operations as discussed by Thorpe [32]. A protograph may also be considered as an $[n', k']$ linear code $\mathscr{P}$ of small block length and dimension. A longer code may be obtained by expanding code $\mathscr{P}$ by an integer factor Q so that the resulting code has parameter $[n = n'Q, k = k'Q]$ over the same field. A simplest way to expand code $\mathscr{P}$ and also to impose structure in the resulting code is by replacing a non-zero element of the parity-check matrix of code $\mathscr{P}$ with a $Q \times Q$ permutation matrix, and a zero element with a $Q \times Q$ zero matrix. As a consequence, the resulting code has a quasi-cyclic structure. The procedure is described in detail in the following example.

Example 12.5 Consider a code $\mathscr{P} = [4, 2]$ over $\mathbb{F}_2$ as a protograph. The parity-check matrix of code $\mathscr{P}$ is given by

$$H' = \begin{array}{c} \\ c_0 \\ c_1 \end{array} \begin{array}{c} \begin{array}{cccc} v_0 & v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \end{array}. \tag{12.13}$$

Let the expansion factor $Q = 5$, the expanded code, which is a [20, 10] code, has a parity-check matrix given by

$$
H = \begin{array}{c|cccccccccccccccccc}
 & v_0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 & v_9 & v_{10} & v_{11} & v_{12} & v_{13} & v_{14} & v_{15} & v_{16} & v_{17} & v_{18} & v_{19} \\
\hline
c_0 & \mathbf{1} & & & & & \mathbf{1} & & & & & & & & & & \mathbf{1} & & & & \\
c_1 & & 1 & & & & & 1 & & & & & & & & & & 1 & & & \\
c_2 & & & 1 & & & & & 1 & & & & & & & & & & 1 & & \\
c_3 & & & & 1 & & & & & 1 & & & & & & & & & & 1 & \\
c_4 & & & & & 1 & & & & & 1 & & & & & & & & & & 1 \\
\hline
c_5 & & & & & & \mathbf{1} & & & & & \mathbf{1} & & & & & \mathbf{1} & & & & \\
c_6 & & & & & & & 1 & & & & & 1 & & & & & 1 & & & \\
c_7 & & & & & & & & 1 & & & & & 1 & & & & & 1 & & \\
c_8 & & & & & & & & & 1 & & & & & 1 & & & & & 1 & \\
c_9 & & & & & & & & & & 1 & & & & & 1 & & & & & 1
\end{array}, \tag{12.14}
$$

where the zero elements have been omitted. This protograph construction may also be described using the Tanner graph representation as shown in Fig. 12.11.

Initially, the Tanner graph of code $\mathscr{P}$ is replicated Q times. The edges of these replicated Tanner graphs are then permuted. The edges may be permuted in many ways and in this particular example, we want the permutation to produce a code which has quasi-cyclic structure. The edges shown in bold in Fig. 12.11 or equivalently the non-zeros shown in bold in (12.14) represent the code $\mathscr{P}$.

The minimum Hamming distance of code $\mathscr{P}$ is 2 and this may be seen from its parity-check matrix, (12.13), where the summation of two column vectors, those of v_1 and v_3, produces a zero vector. Since, in the expansion, only identity matrices are

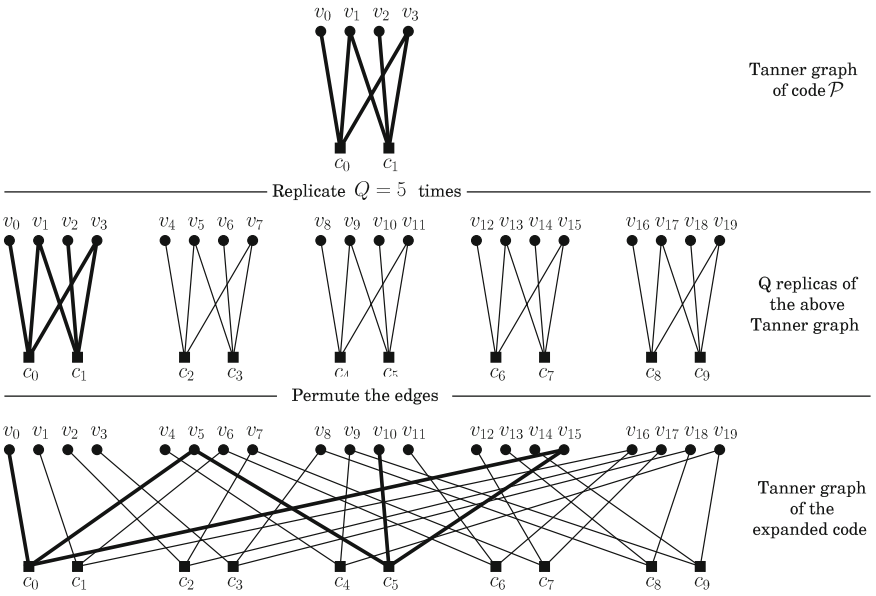


Fig. 12.11 Code construction using a protograph

employed, the expanded code will have the same minimum Hamming distance as the protograph code. This is obvious from (12.14) where the summation of two column vectors, those of v_5 and v_{15}, produces a zero vector. In order to avoid the expanded code having low minimum Hamming distance, permutation matrices may be used instead and the parity-check matrix of the expanded code is given by (12.15).

$$H = \begin{array}{c|cccccccccccccccccc} & v_0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 & v_9 & v_{10} & v_{11} & v_{12} & v_{13} & v_{14} & v_{15} & v_{16} & v_{17} & v_{18} & v_{19} \\ \hline c_0 & & & \mathbf{1} & & & & & & & \mathbf{1} & & & & & & \mathbf{1} & & & & \\ c_1 & & & & 1 & & 1 & & & & & & & & & & & 1 & & & \\ c_2 & & & & & 1 & & 1 & & & & & & & & & & & 1 & & \\ c_3 & 1 & & & & & & & 1 & & & & & & & & & & & 1 & \\ c_4 & & 1 & & & & & & & 1 & & & & & & & & & & & 1 \\ \hline c_5 & & & & & & \mathbf{1} & & & & & & & & \mathbf{1} & & \mathbf{1} & & & & \\ c_6 & & & & & & & & 1 & & & & & & & 1 & & 1 & & & \\ c_7 & & & & & & & & & 1 & & 1 & & & & & & & 1 & & \\ c_8 & & & & & & & & & & 1 & & 1 & & & & & & & & 1 \\ c_9 & & & & & & 1 & & & & & & & 1 & & & 1 & & & & \end{array},$$

(12.15)

The code defined by this parity-check matrix has minimum Hamming distance of 3. In addition, the cycle structure of the protograph is also preserved in the expanded code if only identity matrices are used for expansion. Since the protograph is such a small code, the variable vertex degree distribution required to design a good target code, which has much larger size than a protograph does, in general, causes many inevitable short cycles in the protograph. Using appropriate permutation matrices in the expansion, these short cycles may be avoided in the expanded code.

In the following, we describe a construction of a long quasi-cyclic LDPC code for application in satellite communications. The standard for digital video broadcasting (DVB), which is commonly known as DVB-S2, makes use of a concatenation of LDPC and BCH codes to protect the video stream. The parity-check matrices of DVB-S2 LDPC codes contain a zigzag matrix for the $n-k$ parity coordinates and quasi-cyclic matrices on the remaining k coordinates. In the literature, the code with this structure is commonly known as the irregular repeat accumulate (IRA) code [12].

The code construction described below, using a protograph and greedy PEG expansion, is aimed at improving the performance compared to the rate 3/4 DVB-S2 LDPC code of block length 64800 bits. Let the [64800, 48600] LDPC code that we will construct be denoted by $\mathscr{C}_1$. A protograph code, which has parameter [540, 405], is constructed using the PEG algorithm with a good variable vertex degree distributions obtained from Urbanke [34],

$$\begin{aligned}\Lambda_{\lambda_1}(x) = &\underbrace{0.00185185x + 0.248148x^2}_{\text{for zigzag matrix}} + 0.55x^3 + 0.0592593x^5 \\ &+ 0.0925926x^8 + 0.00555556x^{12} + 0.00185185x^{15} + 0.0166667x^{19} \\ &+ 0.00185185x^{24} + 0.00185185x^{28} + 0.0203704x^{35}.\end{aligned}$$

The constructed [540, 405] protograph code has a parity-check matrix $\boldsymbol{H}' = [\boldsymbol{H}'_u \mid \boldsymbol{H}'_p]$ where $\boldsymbol{H}'_p$ is a 135×135 zigzag matrix, see (12.10), and $\boldsymbol{H}'_u$ is an irregular matrix satisfying $\Lambda_{\lambda_1}(x)$ above. In order to construct a [64800, 48600] LDPC code $\mathscr{C}_1$, we need to expand the protograph code by a factor of $Q = 120$. In expanding the protograph code, we apply the greedy approach to construct the offset matrix $\boldsymbol{O}$ in order to obtain a Tanner graph for the [64800, 48600] LDPC code $\mathscr{C}_1$, which has local girth maximised. This greedy approach examines all offset values, from 0 to $Q - 1$, and picks an offset that results in highest girth or if there is more than one choice, one of these is randomly chosen. A 16200×48600 matrix $\boldsymbol{H}_u$ can be easily constructed by replacing a non-zero element at coordinate (i, j) in $\boldsymbol{H}'_u$ with a permutation matrix $\boldsymbol{P}_{Q,O_{i,j}}$. The resulting LDPC code $\mathscr{C}_1$ has a parity-check matrix given by $\boldsymbol{H} = [\boldsymbol{H}_u \mid \boldsymbol{H}_p]$, where, as before, $\boldsymbol{H}_p$ is given by (12.10).

In comparison, the rate 3/4 LDPC code of block length 64800 bits specified in the DVB-S2 standard takes a lower Q value, $Q = 45$. The protograph is a [1440, 1080] code which has the following variable vertex degree distributions

$$\Lambda_{\lambda_2}(x) = \underbrace{0.000694x + 0.249306x^2}_{\text{for zigzag matrix}} + 0.666667x^3 + 0.083333x^{12}.$$

For convenience, we denote the DVB-S2 LDPC code by $\mathscr{C}_2$.

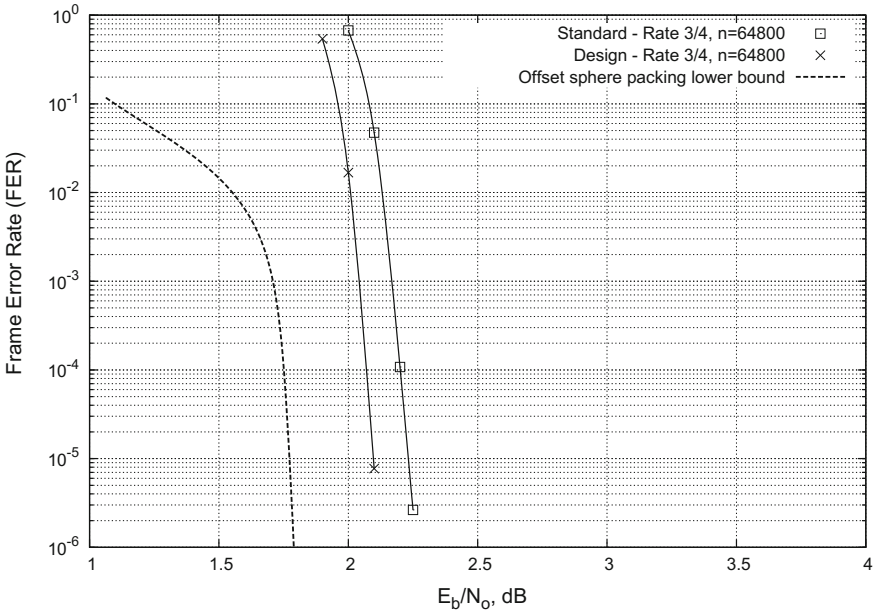


Fig. 12.12 FER performance of the DVB-S2 and the designed [64800, 48600] LDPC codes

Figure 12.12 compares the FER performance of $\mathscr{C}_1$ and $\mathscr{C}_2$ using the belief propagation decoder with 100 iterations. Binary antipodal signalling and AWGN channel are assumed. Note that, although the outer concatenation of BCH code is not used, there is still no sign of an error floor at FER as low as 10^{-6} which means that the BCH code is no longer required. It may be seen from Fig. 12.12 that the designed LDPC code, which at 10^{-5} FER performs approximately 0.35 dB away from the sphere packing lower bound offset for binary transmission loss, is 0.1 dB better than the DVB-S2 code.

12.5 Summary

The application of cyclotomic cosets, idempotents and Mattson–Solomon polynomials has been shown to produce many binary cyclic LDPC codes whose parity-check equations are orthogonal in each position. Whilst some of these excellent cyclic codes have the same parameters as the known class of finite geometry codes, other codes are new. A key feature of this construction technique is the incremental approach to the minimum Hamming distance and the sparseness of the resulting parity-check matrix of the code. Binary cyclic LDPC codes may also be constructed by considering idempotents in the Mattson–Solomon domain. This approach has provided a different insight into the cyclotomic coset-based construction. It has also been shown that, for short algebraic LDPC codes, the myths of codes which have cycles of length 4 in their Tanner graph do not converge well with iterative decoding is not necessarily true. It has been demonstrated that the cyclotomic coset-based construction can be easily extended to produce good non-binary algebraic LDPC codes.

Good irregular LDPC codes may be constructed using the progressive edge-growth algorithm. This algorithm adds edges to the variable and check vertices in a way that maximises the local girth. Many code results have been presented showing the effects of choosing different degree distributions. Guidelines are given for designing the best codes.

Methods of producing structured LDPC codes, such as those which have quasi-cyclic structure, have been described. These are of interest to industry due to the simplification of the encoder and decoder. An example of such a construction to produce a (64800, 48600) LDPC code, using a protograph, has been presented along with performance results using iterative decoding. Better results are obtained with this code than the (64800, 48600) LDPC code used in the DVB-S2 standard.

References

1. Campello, J., Modha, D.S., Rajagopalan, S.: Designing LDPC codes using bit-filling. In: Proceedings of the IEEE ICC, pp. 55–59 (2001)
2. Campello, J., Modha, D.S.: Extended bit-filling and LDPC code design. In: Proceedings of the IEEE Globecom Conference, pp. 985–989 (2001)
3. Chung, S.Y., Forney Jr., G.D., Richardson, T.J., Urbanke, R.L.: On the design of low-density parity check codes within 0.0045 db of the shannon limit. IEEE Commun. Lett. **3**(2), 58–60 (2001)
4. Chung, S.Y., Richardson, T.J., Urbanke, R.L.: Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation. IEEE Trans. Inf. Theory **47**(2), 657–670 (2001)
5. Costello, Jr. D., Forney, Jr. G.: Channel coding: the road to channel capacity (2006). Preprint available at http://arxiv.org/abs/cs/0611112
6. Davey, M.C., MacKay, D.J.C.: Low-density parity-check codes over GF(q). IEEE Commun. Lett. **2**, 165–167 (1998)
7. Etzion, T., Trachtenberg, A., Vardy, A.: Which codes have cycle-free tanner graphs? IEEE Trans. Inf. Theory **45**(6), 2173–2181 (1999)
8. Gallager, R.: Low-density parity-check codes. IRE Trans. Inf. Theory IT **8**, 21–28 (1962)
9. Gallager, R.: Low-Density Parity-Check Codes. MIT Press, Cambridge (1963)
10. Hu, X.Y., Eleftheriou, E., Arnold, D.M.: Irregular progressive edge-growth tanner graphs. In: Proceedings of IEEE International Symposium on Information Theory (ISIT), Lausanne, Switzerland (2002)
11. Hu, X.Y., Eleftheriou, E., Arnold, D.M.: Regular and irregular progressive edge-growth tanner graphs. IEEE Trans. Inf. Theory **51**(1), 386–398 (2005)
12. Jin, H., Khandekar, A., McEliece, R.J.: Irregular repeat-accumulate codes. In: Proceedings of 2nd International Symposium on Turbo Codes and Related Topics, Brest, France, pp. 1–8 (2000)
13. Johnson, S.: Low-Density Parity-Check Codes from Combinatorial Designs. Ph.D dissertation, School of Electrical Engineering and Computer Science, University of Newcastle, Callaghan, NSW 2308, Australia (2004)
14. Johnson, S.J., Weller, S.R.: Construction of low-density parity-check codes from Kirkman triple systems. In: Proceedings of IEEE Information Theory Workshop, Cairns, Australia, 2–7 Sept, pp. 90–92 (2001)
15. Johnson, S.J., Weller, S.R.: Codes for iterative decoding from partial geometries. In: Proceedings IEEE International Symposium on Information Theory, Lausanne, Switzerland, 30 June-3 July p. 310 (2002)
16. Kou, Y., Lin, S., Fossorier, M.: Low-density parity-check codes based on finite geometries: a rediscovery and new results. IEEE Trans. Inf. Theory **47**(7), 2711–2736 (2001)
17. Lin, S., Costello Jr., D.J.: Error Control Coding: Fundamentals and Applications, 2nd edn. Pearson Education, Inc, NJ (2004)
18. Luby, M.G., Shokrolloahi, M.A., Mizenmacher, M., Spielman, D.A.: Improved low-density parity-check codes using irregular graphs. IEEE Trans. Inf. Theory **47**(2), 585–598 (2001)
19. Lucas, R., Fossorier, M.P.C., Kou, Y., Lin, S.: Iterative decoding of one-step majority logic decodable codes based on belief propagation. IEEE Trans. Commun. **46**(6), 931–937 (2000)
20. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North-Holland, Amsterdam (1977)
21. Margulis, G.A.: Explicit constructions of graphs without short cycles and low density codes. Combinatorica **2**(1), 71–78 (1982)
22. McEliece, R.J., MacKay, D.J.C., Cheng, J.F.: Turbo decoding as an instance of pearl's "belief propagation" algorithm. IEEE J. Sel. Areas Commun. **16**, 140–152 (1998)
23. Papagiannis, E., Ambroze, M.A., Tomlinson, M.: Analysis of non convergence blocks at low and moderate SNR in SCC turbo schemes. In: SPSC 2003 8th International workshop on Signal Processing for Space Communications. Catania, Italy, pp. 121–128 (2003)

24. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Mateo (1988)
25. Peterson, W., Weldon Jr., E.J.: Error-Correcting Codes. MIT Press, Cambridge (1972)
26. Ping, L., Leung, W.K., Phamdo, N.: Low density parity check codes with semi-random parity check matrix. Electron. Lett. **35**(1), 38–39 (1999)
27. Richardson, T.J., Shokrollahi, M.A., Urbanke, R.L.: Design of capacity-approaching irregular low-density parity-check codes. IEEE Trans. Inf. Theory **47**(2), 619–637 (2001)
28. Richardson, T.J., Urbanke, R.L.: The capacity of low-density parity-check codes under message-passing decoding. IEEE Trans. Inf. Theory **47**(2), 599–618 (2001)
29. Richter, G., Hof, A.: On a construction method of irregular LDPC codes without small stopping sets. In: Proceedings of IEEE International Conference on Communications, Istanbul, Turkey, pp. 1119–1124 (2006)
30. Shannon, C.E.: A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948)
31. Tang, H., Xu, J., Lin, S., Abdel-Ghaffar, K.A.S.: Codes on finite geometries. IEEE Trans. Inf. Theory **51**(2), 572–596 (2005)
32. Thorpe, J.: Low-density parity-check (LDPC) codes constructed from protographs. JPL IPN Progress Report 42–154 (2003). Available: http://tmo.jpl.nasa.gov/progress_report/42-154/154C.pdf
33. Tian, T., Jones, C., Villasenor, J., Wesel, R.: Selective avoidance of cycles in irregular LDPC code construction. IEEE Trans. Commun. **52**, 1242–1247 (2004)
34. Urbanke, R.: LdpcOpt a fast and accurate degree distribution optimizer for LDPC code ensembles (2001). Available at http://lthcwww.epfl.ch/research/ldpcopt/
35. Vasic, B., Milenkovic, M.: Combinatorial constructions of low-density parity-check codes for iterative decoding. IEEE Trans. Inf. Theory **50**(6), 1156–1176 (2004)
36. Weldon Jr., E.J.: Difference-set cyclic codes. Bell Syst. Tech. J. **45**, 1045–1055 (1966)