LDPC Codes for Communication Systems: Coding Theoretic Perspective

Takayuki NOZAKI\(^{†‡a}\), Member and Motohiko ISAKA\(^{†‡b}\), Senior Member

SUMMARY  
Low-density parity-check (LDPC) codes are widely used in communication systems for their high error-correcting performance. This survey introduces the elements of LDPC codes: decoding algorithms, code construction, encoding algorithms, and several classes of LDPC codes. 

key words: error-correcting code, low-density parity-check code

1. Introduction

Low-density parity-check (LDPC) codes [1, 2] are linear codes defined by a sparse parity check matrix. The LDPC codes are efficiently decoded with complexity $O(n)$ by an iterative decoding algorithm, e.g., sum-product algorithm. The decoding error rates of well-designed LDPC codes approach the channel capacity under the sum-product algorithm [3]. Roughly speaking, LDPC codes have good decoding performance with small decoding complexity.

Nowadays, LDPC codes have been adopted by many digital communication standards: IEEE 802.3 (10GBASE-T) [4], IEEE 802.11 (wifi) [5], IEEE 802.16 (WiMAX) [6], DVB-C2 [7], DVB-S2 [8], DVB-T2 [9], ATSC 3.0 [10], ISDB-S3 [11], CCSDS [12], and 5G New Radio [13].

This survey introduces the elements of LDPC codes and quasi-cyclic LDPC (QC-LDPC) codes, which are a class of LDPC codes used in modern communication systems. Moreover, we briefly explain several classes of LDPC codes which may be used in future communication systems.

Section 2 introduces the fundamentals of error-correcting codes, and the definition and properties of LDPC codes. We explain the elements of LDPC codes: decoding algorithms (Sect. 3), construction (Sect. 4), and encoding algorithms (Sect. 5). Next, we describe two classes of LDPC codes called protograph LDPC codes (Sect. 6) and QC-LDPC codes (Sect. 7); QC-LDPC codes are adopted in many digital communication standards and can be clearly described in the context of protograph LDPC codes. Additionally, we explain spatially coupled LDPC codes (Sect. 8) and non-binary LDPC codes (Sect. 9) since those classes have good decoding performance and may be used in the future. Section 10 provides parameters of LDPC codes used in some communication standards. Finally, Sect. 11 summarizes the survey.

2. Preliminaries

Almost all the codes in practical use are linear codes over finite fields $\mathbb{F}_q$ with $q = 2^m$ for a positive integer $m$. Binary linear codes over $\mathbb{F}_2 = \{0, 1\}$ are especially of interest, and this survey focuses on them unless otherwise stated.

2.1 System Model, Channel Capacity and Coding

Suppose that an information vector that is represented by $u = (u_1, \ldots, u_k) \in \mathbb{F}_2^k$ is to be transferred from a sender to a receiver. The encoder is usually an injective mapping from $u$ to a codeword $c = (c_1, \ldots, c_n) \in C$, where $C$ is called a code and is a subset of $\mathbb{F}_2^n$. The codeword $c$ may be directly transmitted to the channel, or as a modulated signal sequence $x = (x_1, \ldots, x_n) \in X$, where we assume $X = \{\pm 1\}$ is a binary channel input alphabet. The rate of this coding scheme is the number of information bits transferred per channel use, and is defined as $R = \log_2 2^k / n = k / n$.

The decoder determines the estimate for $\hat{c}$ (equivalently $\hat{u}$) for $c$ ($u$) from the received vector $y$. Word error probability (WEP) is the probability that the estimate of $\hat{c}$ does not coincide with the original codeword $c$. Block-wise maximum \textit{a posteriori} (MAP) decoding $\hat{c} = \arg \max_{c \in C} P(c | y)$ minimizes WEP, and is equivalent to maximum-likelihood (ML) decoding $\hat{c} = \arg \max_{c \in C} P(y | c)$ when codewords follow the uniform distribution. Symbol error probability (SEP) (or bit error probability (BEP) for $q = 2$) is the probability of $\hat{c}_i \neq c_i$ averaged over all indices $i = 1, \ldots, n$. Minimum SEP is achieved by symbol-wise MAP decoding in which estimate of each symbol depends on the marginalized \textit{a posteriori} probability: $\hat{c}_i = \arg \max_{c_i \in \mathbb{F}_2} P(c_i | y) = \arg \max_{c_i \in \mathbb{F}_2} \sum_{c \in C, c_i = c_i} P(c | y)$ for $i = 1, \ldots, n$. However, these decoding criteria in general require exponentially large computational cost in $n$, and more efficient approaches need to be explored.
Consider for simplicity a memoryless channel which
is characterized by the conditional probability distribution
\(P_{Y|X}\) of channel output \(Y\) given channel input \(X\). It is
practically reasonable to assume that \(X\) follows the uniform
distribution on the channel input alphabet \(\mathcal{X}\). The input-
constrained capacity of a channel with respect to \(\mathcal{X}\) is the
mutual information \(I(X;Y)\), and is the supremum of achievable
rate at which there exists a sequence of coding and de-
coding schemes with vanishing error probability [14]. One
of the ultimate goals in coding theory is thus to explicitly
design encoder and decoder pair achieving or approaching
the capacity. For instance, on the binary-input additive white Gaussian noise
channel (BEC), each transmitted binary symbol is erased with prob-
ability \(p\) and it is received correctly otherwise. The capacity is
1 – \(p\), and accordingly reliable communication is possible
by properly designing codes of possibly large \(n\) if \(R < 1 – p\).

On the binary-input additive white Gaussian noise channel
(BI-AWGN) in which \(Y = X + Z\), \(X \in \{\pm 1\}\), and
\(Z \sim \mathcal{N}(0, \sigma^2)\) is the Gaussian noise, reliable commu-
nication of \(R = 1/2\) is possible at \(E_b/N_0 = 1/(2\sigma^2 R) > 0.2\) (dB)
whereas BEP=10^{-5} is observed at 9.6 dB for the uncoded
transmission as will be seen in Sect. 2.4.

2.2 Linear Codes over Finite Field

An \((n, k, d_{\text{min}})\) binary linear code \(C\) is defined as a \(k\)-
dimensional subspace of \(\mathbb{F}_2^n\), and each vector in \(C\) is called a
codeword. Minimum distance \(d_{\text{min}}\) of \(C\) is the smallest Ham-
ing distance between two distinct codewords in \(C\), and is
usually an important parameter with respect to error perfor-
mance at low error rates. For linear codes, \(d_{\text{min}}\) is equal to the
smallest Hamming weight of any non-zero codewords.

For a given set of basis vectors \(\{g_1, \ldots, g_k\}\) in \(C\), where
\(g_i = (g_{1i}, \ldots, g_{ni})\), any codeword is their linear combina-
tion \(c = u_1g_1 + \cdots + u_6g_6\), which can be regarded as an
encoding map from the information vector \(u\) to \(c\). It is equiv-
antly expressed as \(c = uG\), where \(k \times n\) matrix \(G = \{g_{i,j}\}\)
is called a generator matrix of \(C\). Note that encoding map is
not unique for a given linear code, and is called systematic
when information symbols are in a codeword or \(c_j = u_i\) for
\(i = 1, \ldots, k\) and some \(j \in \{1, \ldots, n\}\).

The dual code of \(C\) is defined as \(C^\perp = \{v : vc^T = 0, c \in C\}\),
namely the set of all the vectors which are orthogonal to
any codeword in \(C\). This is indeed an \((n-k)\)-dimensional
subspace of \(\mathbb{F}_2^n\) with a set of basis vectors \(\{h_1, \ldots, h_{n-k}\}\),
where \(h_i = (h_{i1}, \ldots, h_{in})\). An \((n-k)\times n\) matrix \(H = \{h_{i,j}\}\)
is called a parity-check matrix of \(C\), and \(GH^T\) is a matrix of
all-zero elements by definition. It follows that designing a
linear code \(C\) over \(\mathbb{F}_2\) is reduced to specifying either \(G\) or \(H\).

A vector \(v \in \mathbb{F}_2^n\) is a codeword of \(C\) if and only if its
syndrome \(s = Hv^T\) is an all-zero vector \(0 = (0, \cdots, 0)\). Each
parity-check equation \(h_1c^T = 0\), \(i = 1, \ldots, n - k\) for \(c \in C\)
represents a local constraint imposed on the code in the sense
that weighted sum of the symbols (exclusive-OR in the binary
case) in a codeword must be zero. Decoding for LDPC codes
in Sect. 3 iteratively performs computation on such
local constraints, aiming at determining codeword symbols
subject to the global constraint of all-zero syndrome.

The linear codes are also defined by bipartite graphs
called Tanner graphs. A Tanner graph \(G = (V \cup C,E)\) consists of
two sets of nodes, namely the set \(V\) of variable nodes and
the set \(C\) of check nodes, and the set \(E\) of edges. Each parity
check matrix is defined from a Tanner graph; Set \(h_{i,j} = 1\)
if and only if the \(i\)-th check node \(c_j\) is adjacent to the \(j\)-th
variable node \(v_j\). The \(j\)-th variable node associates to the
\(j\)-th symbol of a codeword. The \(i\)-th check node represents
the constraint given by the \(i\)-th row of parity check matrix,
i.e., \(h_{i}c^T = 0\). The degree of a node is the number of edges
connecting to the node. The degree of the \(j\)-th variable
node (resp. \(i\)-th check node) equals the number of non-zero
elements in the \(j\)-th column (resp. \(i\)-th row).

**Example 2.1:** Figure 2 depicts an example of a Tanner
graph. The circles (resp. squares) in Fig. 2 represent the vari-
able (resp. check) nodes. The corresponding parity check
matrix is

\[
\begin{bmatrix}
1 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 \\
\end{bmatrix}
\]

Linear codes, a tiny subset of all possible codes over \(\mathbb{F}_2^n\),
greatly facilitate the code description through \(G\) or \(H\) which
also enables encoding by linear operation. On the other hand,
decoding for general linear codes still require exponentially
large computational cost in \(n\) at least when (near-)optimum
performance is pursued. Accordingly, particular (class of)
linear codes with certain structure for efficient decoding have
been studied as outlined in Sect. 2.4. In this sense, the impact
of LDPC codes (and turbo codes in Sect. 2.4.2) was more
significant with respect to decoding whose complexity is
basically linear in \(n\).

2.3 Definition of LDPC Code

An LDPC code is defined by a sparse parity check matrix
\(H \in \mathbb{F}_q^{(n-k)\times n}\) as follows:

\[
C = \{c \in \mathbb{F}_q^n : Hc^T = 0^T\}
\]

It is called a binary (resp. non-binary) LDPC code if \(q = 2\)
(resp. \(q > 2\)). Except in Sect. 9, we only consider the binary
LDPC codes.

The \((c,d)\)-regular LDPC codes are defined by the Tanner
graphs such that every variable node has degree \(c\) and every
check node has degree \(d\). The irregular LDPC codes are
defined by the Tanner graphs of which degrees of nodes are
chosen according to a pair of degree distributions. The degree distribution \( \lambda(x) = \sum_i \lambda_i x^{-i} \) (resp. \( \rho(x) = \sum_i \rho_i x^{-i} \)) characterizes the degrees of variable (resp. check) nodes, where \( \lambda_i \) (resp. \( \rho_i \)) denotes the fraction of edges connecting to the variable (resp. check) nodes of degree \( i \). For example, the degree distribution pair of the code in Example 2.1 is \( \lambda(x) = \frac{4}{7} x + \frac{3}{7} x^2 \) and \( \rho(x) = \frac{3}{7} x^2 + \frac{4}{7} x^3 \).

2.4 LDPC and Other Codes

In this subsection, some important codes in coding theory and its practice are reviewed partly with a purpose of addressing the properties of LDPC codes.

2.4.1 BCH and Reed-Solomon Codes

The proof for direct part of the channel coding theorem was not constructive and relies on the notion of random coding [14], which unfortunately requires prohibitively large decoding complexity for large \( n \). Algebraic coding theory [15] was intensively studied for the first few decades as it allows efficient encoding and decoding based on the inherent structure, including binary BCH codes, Reed-Solomon (RS) codes over \( F_q \), as well as Reed-Muller (RM) codes. In particular, RS codes have been used in optical discs and many other applications.

BCH and RS codes ensure large \( d_{\text{min}} \) for given \( n \) and \( k \) when code length is limited, and are advantageous in terms of error probability in such a scenario when ML decoding is performed [16]. However, algorithms based on the algebraic structure are mostly hard-decision bounded-distance decoding in which received signals are quantized to symbols in \( F_q \) at decoder input and corrects up to \( \lfloor (d_{\text{min}} - 1)/2 \rfloor \) symbol errors. Efficient near-ML decoding has not been available for large \( n \) especially for soft-decision decoding on continuous-output channels.

Binary BCH codes are used as outer code for concatenation with inner LDPC code in recent broadcasting systems [7]–[11]. The role of the outer code is in handling residual errors at the output of LDPC decoder, and is effective when the latter suffers from relatively low weight error events in the error floor region to be mentioned below.

The sum-product algorithm for LDPC codes in Sect. 3.1 performs poorly when it is applied to BCH codes as the parity-check matrices are not sparse. Neural belief propagation (BP) decoding is recently proposed to alleviate this problem by training weights incorporated in the decoding formula on the neural network obtained by unrolling the Tanner graph [17].

2.4.2 Turbo Codes

It was just before the rediscovery of LDPC codes [18] when efficiently decodable near-capacity coding scheme called turbo codes was proposed in 1993 [19], and it has been subsequently used in third and fourth mobile communication systems [20]. These codes are distinct from conventional schemes in that (i) simple codes are combined in a pseudorandom manner, and (ii) low-complexity iterative decoding is available even for large block lengths [21]. Indeed, turbo codes are parallel concatenation of two convolutional encoders, while LDPC codes can be regarded as consisting of repetition and single-parity check codes.

Figure 3 shows what coding theory encountered in the mid 1990’s: bit error rate (BER) of (32,16,8) RM code, (128,64,22) extended BCH code, \( R = 1/2 \) turbo code of \( n = 4000, 20000 \), (2, 4)-regular LDPC code of \( n = 500, 2500 \) over \( F_q \) (4000 and 20000 bits long) over the BI-AWGN.

Error probability of both turbo and LDPC codes under iterative decoding typically exhibits the following two regions: waterfall region in which error probability drops at certain channel parameter \( E_b/N_0 \) in the case of Fig. 3, more sharply for larger block length \( n \), and is a direct consequence of the behavior of iterative decoding; error floor region in which error probability decreases more slowly and is observed when near-optimum decoding for the code is achieved by iterative decoder. In addition, decoding failure due to certain topology in a Tanner graph may cause the error floor for LDPC codes as stated in Sect. 4.

The error floor observed for BER of the turbo code in Fig. 3 is due to small \( d_{\text{min}} \), and it performs poorly with respect to WEP. In contrast, \( d_{\text{min}} \) of LDPC codes tend to be larger and typical minimum distance of regular LDPC codes grows linearly in \( n \) [2].

The underlying principle of decoding is shared by these codes, and is an instance of more general framework of the sum-product algorithm [23]. Symbol-wise MAP decoding is essentially performed at each decoder associated with the two component codes. We may write

\[
P(x|y) \propto P(y|x)P(x) = P(y|x) \prod_{i=1}^n P(x_i),
\]

where independence of the codeword symbols is assumed. The \( a \ priori \)
probability \( P(c_i) \) for the symbol \( c_i \) is provided as extrinsic information from the other decoder, which in turn performs symbol-wise MAP decoding but setting the \textit{a priori} probability of the corresponding symbol as 0.5 [24]. This exchange of extrinsic information between two decoders is iterated until it finds a codeword or up to predefined number of iterations. Decoding for LDPC codes in Sect. 3 follows a similar line at variable nodes (repetition codes) and check nodes (single parity-check codes) on a Tanner graph.

2.4.3 Polar and RM Codes

Polar codes proposed by Arikan are provably capacity-achieving for binary-input symmetric discrete channels [25]. Rows of generator matrix \( \mathbf{G} \) for polar codes of length \( n = 2^m \) are subset of those in the Kronecker product \( \mathbf{F}^\otimes m \) of \( \mathbf{F} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \). In the successive cancellation (SC) decoder, the decision for each symbol in the information vector \( \mathbf{u} \) is made in order under the assumption that the already decoded estimate is correct. By a proper choice of \( \mathbf{G} \) and decoding order, virtual \( k = nR \) channels associated with information symbols tend toward noise-free and thus results in capacity-achieving performance for asymptotically large \( n \) [26]. However, decrease of error probability in \( n \) is rather slow under SC decoding. CRC-assisted list decoding was proposed to overcome this problem in [27], and polar codes are consequently competitive with LDPC codes even for practical range of \( n \). Together with such progress, polar codes are deployed for the control channel in 5G New Radio [13]. BP decoding in Sect. 3.1 are also discussed for polar codes in [28] and subsequent works.

We note that RM codes attract renewed interests motivated by the success of polar codes, as generator matrices of RM codes with \( n = 2^m \) consist of the rows of \( k \) largest weights in \( \mathbf{F}^\otimes m \). Recent results include the proof of optimality on the BEC [29] and near-ML performance with SC or BP decoding for \( n = 128 \) RM codes by parallel decoding based on automorphism group of the code [30].

3. Decoding of LDPC Codes

We start with decoding for LDPC codes due to its importance as mentioned in Sect. 2.2, and introduce encoding as well as code designs in later sections.

Basically, LDPC codes are decoded by a message passing algorithm, which is a class of iterative algorithms. The message passing algorithms work on the Tanner graphs of LDPC codes. In the message passing algorithms, messages are calculated on the nodes and conveyed by edges. Each message represents an estimation of the bit of the associated variable node.

This section mainly describes two message passing algorithms, called sum-product (SP) algorithm and bit-flipping (BF) algorithm, both introduced by Gallager [1]. This section also provides some improved or simplified versions of the SP algorithm in Sect. 3.3. The SP and BF algorithms have decoding complexity of \( \mathcal{O}(n) \), which is much smaller than that of ML decoding or MAP decoding. Moreover, both algorithms are parallel, i.e., each node calculates outgoing messages independently of each other.

Let \( \mathcal{N}_v(i) \) (resp. \( \mathcal{N}_c(i) \)) be the set of indices of the check (resp. variable) nodes which is adjacent to the \( i \)-th variable (resp. check) node, i.e., \( \mathcal{N}_v(i) := \{ j : h_{i,j} = 1 \} \) (resp. \( \mathcal{N}_c(i) := \{ j : h_{i,j} = 1 \} \)). We write \( \mu_{j \rightarrow i} \) for the message from the \( j \)-th variable node to the \( i \)-th check node and \( v_{i \rightarrow j} \) for the message from the \( i \)-th check node to the \( j \)-th variable node.

3.1 Sum-Product Algorithm

The SP algorithm is also referred to as the belief propagation (BP) algorithm. This algorithm is a soft-decision decoding algorithm for the LDPC codes.

Suppose that a codeword \( \mathbf{c} \) is modulated to a signal sequence \( \mathbf{x} \in \{ \pm 1 \}^n \), where

\[
x_i = \begin{cases} +1, & \text{if } c_i = 0, \\ -1, & \text{if } c_i = 1. 
\end{cases}
\]

Assume signal sequences are transmitted over the memoryless binary-input output-symmetric channel, whose channel transition probability satisfies

\[
P_{Y|X}(y|x) = \prod_{i=1}^{n} P_{Y|X}(y_i|x_i),
\]

\[
P_{Y|X}(y+1|x) = P_{Y|X}(-y|y - 1).
\]

For a channel transition probability \( P_{Y|X}(y|x) \), we define the log-likelihood ratio (LLR) as

\[
l(y) := \log \frac{P_{Y|X}(y+1)}{P_{Y|X}(y-1)}. \tag{1}
\]

Note that \( l(y) := +\infty \) if \( P_{Y|X}(y | y - 1) = 0 \), and \( l(y) := -\infty \) if \( P_{Y|X}(y+1 | y) = 0 \).

In the SP algorithm, each message takes a value in \( \mathbb{R} \cup \{ \pm \infty \} \) and represents an estimation of the associated variable node. Roughly speaking, the sign of a message gives the estimated symbol value of \( x \), and the absolute value of a message shows reliability of the estimation. Algorithm 1 gives the details. A toy example of the SP algorithm is in [31, pp.118–119].

3.2 Bit-Flipping Algorithm

The bit-flipping (BF) algorithm is a hard-decision message passing algorithm, in which each message takes a value in \( \mathbb{F}_2 \). The BF algorithm is regarded as a quantized SP algorithm, and hence has a lower time/space complexity but has a higher decoding error rate than the SP algorithm.

The BF algorithm has many variations. Hence, we introduce a typical BF algorithm in this survey. Denote the degree of the \( i \)-th variable node, by \( \deg(v_i) \). For each
position \(i\) of variable node and each decoding round \(t\), set two thresholds \(b_{i,t}, d_{i,t}\), where \((\deg(v_{i,t}) - 1)/2 < b_{i,t} \leq \deg(v_{i,t}) - 1\) and \(\deg(v_{i,t})/2 < d_{i,t} \leq \deg(v_{i,t})\).

Algorithm 2 gives the details. Note that the check node calculation satisfies the following property; If the check node constraint is satisfied, i.e., \(\sum_{k \in N(i)} \mu_{k \rightarrow i} = 0\), \(v_{j \rightarrow i} = \mu_{j \rightarrow i}\) holds; Otherwise, \(v_{j \rightarrow i} \neq \mu_{j \rightarrow i}\) holds. In other words, the messages from the check nodes represent whether the check constraints are satisfied. The variable node calculation is transformed as follows:

\[
\mu_{j \rightarrow i} = \begin{cases} 1 + s_j, & \text{if } |\{k \in N(j) \setminus \{i\} : v_{j \rightarrow k} = s_j\}| \geq b_{j,t}, \\ s_j, & \text{otherwise}. \end{cases}
\]

In words, the variable nodes flip their values if they receive many unsatisfied messages.

Careful design of flipping rules in the BF algorithm narrows the performance gap with the SP algorithm. The weighted BF algorithm [32] was proposed for decreasing the decoding error rate of the BF algorithm. Wu et al. [33] showed that the weighted BF algorithm is a simplification of the min-sum algorithm (given in Sect. 3.3.2). Wadayama et al. [34] proposed the gradient descent BF (GDBF) algorithm, which has a higher decoding performance than the weighted BF algorithm. This algorithm decides the flipping rule from the gradient descent for a non-convex optimization problem derived from a decoding problem. Moreover, Wadayama et al. [34] modified the GDBF algorithm with a small noise factor to escape from local optimum points of the optimization problem. Recently, Savin [35] proposed the GDBF algorithm with momentum by considering the past updates.

A simple BF algorithm adopts \(b_{i,t} = \deg(v_{i,t}) - 1\) and \(d_{i,t} = \deg(v_{i,t})\) for all \(t\).
Namely, outgoing message $v_{j-i}$ is simply the incoming message with the smallest absolute value at the $i$-th check node (except the one from $j$-th variable node), while the sign of $v_{j-i}$ remains the same as that in the SP algorithm. Complexity reduction by eliminating $\tanh(\cdot)$ computation is achieved at the cost of some performance degradation which causes by the greater value of $|v_{j-i}|$ than that in the SP algorithm. To remedy this loss, modifications to min-sum algorithm were proposed by Chen et al. in [41] and the references therein. Normalized min-sum algorithm scales the message by multiplying $v_{j-i}$ by a constant $0 < \alpha < 1$. Offset min-sum algorithm alternatively subtracts a constant $\beta > 0$ so that the magnitude of $v_{j-i}$ is replaced by $\max\{\min_{k \in N_i(\mathcal{V})} |\mu_{k-i}| - \beta, 0\}$. 

Other low complexity algorithms are summarized in [42, Sect. 5.5].

3.3.3 Quantization

In the hardware implementation, floating-point numbers for the messages require large space of circuit and power consumption. Hence, the quantization of messages is an important topic. An extreme example of a quantized (or finite alphabet) iterative decoding algorithm is bit-flipping, which uses one-bit messages.

Elaborate non-uniform quantization has better decoding error rates than the uniform quantization (e.g., see [43]). Lee et al. [43] designed the message quantization for the sum-product algorithm based on the mutual information. Chen et al. [41] gave the quantized offset/normalized min-sum algorithm. For the binary symmetric channel, Planjery et al. [44] proposed a finite alphabet iterative decoding algorithm, which is not a mimic of the SP decoding algorithm, and showed that the algorithm has a lower error floor than the SP algorithm.

4. Construction of LDPC Codes

In the construction of LDPC codes, initially, we should improve the decoding error rates in the waterfall region. Next, we lower the decoding error rates in the error floor region.

Each channel model is characterized by some channel parameters, e.g., the channel parameter of the BEC is the channel erasure probability $p$. For simplicity, we assume that a channel model characterized by a single parameter. Then, there exists a supremum of the channel parameter such that the bit decoding error rate goes to 0 as the code length tends to infinity [1]-[3]. This supremum is called decoding threshold [3].

To improve the performance in the waterfall region, we should optimize the macro-structure of the LDPC code, e.g., the pair of degree distribution for the irregular LDPC codes. For the optimization of the macro-structure, we search a code ensemble† with a good decoding threshold. The decoding thresholds are determined by an asymptotic analysis technique called density evolution [3] or its approximation called extrinsic information transfer (EXIT) chart [45]. The decoding error rates in the waterfall regions are also estimated by a finite-length analysis technique called finite-length scaling [46]-[49] for the several code ensembles and channel models. The macro-structure is typically optimized via linear programming or differential evolution.

To lower the error floor region, we should optimize the micro-structure of the LDPC code, i.e., the local structure of the Tanner graph: small cycles, approximate cycle extrinsic message degree (ACE) [50], stopping sets [51], and trapping sets [52]. We should remove such local structures to lower the error floor. Hu et al. [53] proposed progressive edge growth (PEG), a greedy algorithm to construct a Tanner graph without small cycles. This algorithm is extended to removing small ACE [54], small stopping sets [55], and small trapping sets [56]. He et al. [57] proposed an improved PEG algorithm called multi-edge metric-constrained PEG.

5. Encoding of LDPC Codes

One might think that encoding of LDPC codes is easy by using its generator matrix. However, even if the parity check matrix is sparse, corresponding generator matrices are not always sparse. Hence, the encoding algorithm using generator matrices requires $O(n^3)$ complexity in general.

There are two approaches to reduce the encoding complexity for LDPC codes: (i) considering an efficient encoding algorithm for arbitrary LDPC codes and (ii) designing efficiently encodable LDPC codes.

5.1 Efficient Encoding Algorithms

Suppose encoding is systematic in the narrow sense, i.e., the codeword $\epsilon$ is split into the information part $u$ and the parity part $p$ as $\epsilon = (u \mid p)$. Moreover, suppose the parity check matrix is full rank, i.e., $\text{rank}(H) = n - k$.

Encoding algorithms for LDPC codes are divided into two stages: preprocessing stage and encoding stage. The preprocessing stage transforms the parity check matrix $H$ into a systematic form $H_{sys} = (H_0 \mid H_p)$ by arranging the columns and rows of $H$, where $H_0$ is an $(n-k) \times k$ matrix and $H_p$ is an $(n-k) \times (n-k)$ non-singular matrix. Since $H_0^T = 0^T$ holds, we have

$$H_p p^T = H_0 u^T. \quad (2)$$

The encoding stage generates the codeword $\epsilon$ corresponding to a message $u$ by solving Eq. (2). For reducing the encoding complexity, $H_p$ should have a form such that Eq. (2) is efficiently solved.

†Roughly speaking, a code ensemble is a set of codes. For example, the $(3,6)$-regular LDPC code ensemble is the set of all $(3,6)$-regular LDPC codes.

††To keep decoding performance of the code and the sparsity of the parity check matrix, the preprocessing stage transforms $H$ only by arranging the columns and rows, i.e., row and column permutations.
5.1.1 Encoding Algorithm Based on Approximate Triangular Matrix

Richardson and Urbanke [51] proposed the first efficient encoding algorithm. The preprocessing stage transforms $H$ into approximate triangular matrix, described as:

$$H_{\text{ATM}} = \begin{pmatrix} H_{1,a} & T & C \\ H_{1,l} & D & E \end{pmatrix},$$

where $H_1 = \begin{pmatrix} H_{1,a} \\ H_{1,l} \end{pmatrix}$. $H_p = \begin{pmatrix} T \\ D \\ E \end{pmatrix}$. $T$ is an $(n-k-g) \times (n-k-g)$ upper triangular matrix, and the sizes of other matrices are determined accordingly by the size of $T$. Define $\Phi := (E - DT^{-1}C)$ and store $\Phi^{-1}$. Figure 4 illustrates an approximate triangular matrix. We see that the right of this matrix is approximate triangular.

We split parity part $p$ into $(p_1 \mid p_2)$, where $p_1$ and $p_2$ are of length $n-k-g$ and $g$, respectively. The encoding stage is described as follows:

1. Compute $p_2^T = \Phi^{-1}(DT^{-1}H_{1,a} - H_{1,l})u^T$. Here, the details of this step are follows; (i) Calculate $a_1^T = H_{1,a}u^T$; (ii) Solve $a_2^T$ by backward substitution $Ta_2^T = a_1^T$; (iii) Set $a_3^T = Da_2^T$ and $b^T = H_{1,l}u^T$; (iv) Calculate $p_3^T = (a_3^T - b^T)$.

2. Solve $p_1$ by backward substitution; $Tp_1^T = -Cp_2^T - H_{1,a}u^T$.

Note that the backward substitution to sparse triangular matrix requires $O(n)$ complexity. Since the $\Phi^{-1}$ is dense $g \times g$ matrix in general, the multiplication in step 1-(iv) requires $O(g^2)$. It is known [51] that $g$ is proportional to $n$ but small enough. Hence, the encoding complexity of this algorithm is $O(n + g^2)$.

5.1.2 Other Encoding Algorithms

The following algorithms are proposed to reduce the time complexity or encoding time. Kaji [58] considered an encoding algorithm based on LU-factorization. Shibuya [59] proposed an encoding algorithm based on the block-triangularization via Dulmage-Mendelsohn decomposition. Nozaki [60] presented a parallel encoding algorithm based on block-diagonalization. Iketo and Nozaki [61], [62] proposed an encoding algorithm based on block-triangularization. The properties and performance are compared in [62].

5.2 Linear-Time Encodable LDPC Codes

Repeat accumulate (RA) [63] codes and irregular repeat accumulate (IRA) codes [64] are known as sub-classes of LDPC codes with a fast encoding structure. Accumulate repeat accumulate (ARA) codes [65] are also fast encodable LDPC and can be viewed as precoded IRA codes. Those codes are known to be represented by protographs [66] (Sect. 6) but suffer from high error floors.

Quasi-cyclic LDPC (QC-LDPC) codes (Sect. 7) are efficiently encodable by shift-register-adder-accumulator circuits [67].

6. Protograph LDPC Codes

Protograph LDPC codes [66] are a sub-class of LDPC codes, more precisely a sub-class of multi-edge type LDPC codes [68]. Some protograph LDPC codes, e.g., ARA codes, are encodable in $O(n)$ [63]–[65], [69]. Moreover, well-designed protograph LDPC codes, e.g., accumulate repeat jagged-accumulate (ARJA) codes [69], [70], has a good decoding performance in both waterfall and error floor regions.

A protograph is a Tanner graph $(V \cup C, E)$ with a relative small number of nodes, where parallel edges are allowed. The adjacency matrix for a protograph is called a base matrix $B = (b_{i,j})$, where $b_{i,j}$ is a number of edges between $c_i$ and $v_j$. A copy-and-permute operation to protograph provides larger derived graphs. This operation copies the protograph $N$ times and permutes the edges within each bundle, a set of edges generated from the same edge of the protograph. The derived graph becomes the Tanner graph of a protograph LDPC code. In other interpretation, we get a parity check matrix of a protograph LDPC code by replacing the $(i,j)$-th entry of the base matrix into the sum of $b_{i,j}$ random $N \times N$ permutation matrices for each $(i,j)$.

Example 6.1: Figure 5(a) depicts the protograph with a base matrix

$$
\begin{bmatrix}
2 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 
\end{bmatrix}.
$$

By copying four times, we get Fig. 5(b). By permuting each bundle in Fig. 5(b), we obtain a derived graph as in Fig. 5(c). The parity check matrix corresponding to the derived graph in Fig. 5(c) is

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

We see that each $4 \times 4$ sub-matrix of this parity check matrix
Now, let us consider the encoding process of the AR3A code. Let $x_{i,t}$ be the bit stored in $v_{i,t}$, for $i = 1, 2, \ldots, 5$ and $t = 1, 2, \ldots, N$. Suppose $c_{1,i}$ is adjacent to $v_{2,\pi(i)}$. Firstly, assign $2N$ information bits to the variable nodes $v_{1,1}, v_{1,2}, \ldots, v_{1,N}, v_{2,3}, \ldots, v_{2,N}$. Secondly, calculate the bits in $v_{3,1}, v_{3,2}, \ldots, v_{3,N}$. Decide the bit in $v_{3,1}$ by the constraint of $c_{1,1}$: $x_{3,1} = (x_{1,1} + x_{2,\pi(1)})$. Next, compute the bit in $v_{3,2}$ by $c_{1,2}$: $x_{3,2} = (x_{1,2} + x_{2,\pi(2)}) + x_{3,1}$. Similarly, calculate the bits in $v_{3,3}, v_{3,4}, \ldots, v_{3,N}$ by $x_{3,t} = (x_{i,t} + x_{2,\pi(t)}) + x_{3,t-1}$ ($t = 3, 4, \ldots, N$). Thirdly, calculate the $2N$ parity bits, i.e., the bits in $v_{4,1}, v_{4,2}, \ldots, v_{4,N}, v_{5,1}, \ldots, v_{5,N}$. Decide the bit in $v_{4,1}$ by the constraint of $c_{2,1}$. Next, compute the bit in $v_{5,1}$ by $c_{3,1}$. Similarly, calculate the bits in $v_{4,2}, v_{4,3}, \ldots, v_{5,N}$.

Since the bits in $v_{3,1}, v_{3,2}, \ldots, v_{3,N}$ are not transmitted, the number of transmitted bits are $4N$. Hence, the code rate is $1/2$.

Most high performance protograph LDPC codes contain the punctured variable nodes. However, punctured variable nodes require many decoding iterations. Uchikawa [71] designed high performance protograph LDPC codes without punctured nodes.

For more information of the protograph LDPC codes, see [72].

7. Quasi-Cyclic LDPC Codes

Quasi-cyclic (QC) LDPC codes [73] are a special case of protograph LDPC code. QC-LDPC codes have advantages in hardware implementation of the encoder [67] and the decoder. Hence, most of standards (e.g., [5], [6], [13]) adopt QC-LDPC codes.

Define an $N \times N$ cyclic permutation matrix $P$ as

$$
P = \begin{pmatrix}
0 & 0 & \cdots & 0 & 1 \\
1 & 0 & \cdots & 0 & 0 \\
0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 1 & 0
\end{pmatrix}.
$$

For a vector $v$, $v(P)^T$ stands left circular shift of $v$ by $i$ positions. Note that every cyclic permutation matrix is represented by $P$ with some $i$.

The quasi-cyclic codes are defined by the null space of parity check matrices $H = [A_{i,j}]$, where each sub-matrix $A_{i,j}$ is circulant, i.e., the sum of cyclic permutation matrices. The quasi-cyclic code is QC-LDPC if the parity check matrix is sparse. In the terminology of protograph LDPC code, a parity check matrix of a QC-LDPC code is derived by replacing the $(i,j)$-th entry of a base matrix into the sum of $b_{i,j}$ random $N \times N$ cyclic permutation matrix for each $(i,j)$. Hence, QC-LDPC codes are special cases of protograph LDPC codes.

The generator matrices of QC-LDPC codes are also represented by block matrices with circulant matrices. Recall that the multiplication of a cyclic permutation matrix
and a vector \( v \) is a circular shift of \( v \). Hence, the encoder of QC-LDPC codes is implemented by shift registers [67]. Panteleev [74] proposed a fast encoding algorithm for QC-LDPC codes based on the Chinese remainder theorem.

The circulant matrices are also suitable for the hardware implementation of decoders, e.g., a high-throughput FPGA-based decoder shown in [75]. The construction and details for QC-LDPC codes are summarized in [42].

8. Spatially Coupled LDPC Codes

Spatially coupled LDPC (SC-LDPC) codes [76] or LDPC convolutional codes [77] are a sub-class of protograph LDPC codes. This code is constructed by coupling protograph LDPC codes. In this section, we explain the SC-LDPC codes based on protographs.

Let \( B \) be the base matrix of a protograph LDPC code, which is called an un-coupled LDPC code for distinction from an SC-LDPC code. Recall that each entry of the base matrix is an integer. Split \( B \) into \( w \) parts \( B_1, B_2, \ldots, B_w \) as \( B = B_1 + B_2 + \cdots + B_w \). Make the base matrix of the SC-LDPC code as follows:

\[
\begin{pmatrix}
B_0 & O \\
B_1 & B_0 \\
\vdots & \vdots \\
B_w & B_{w-1} \\
O & \cdots
\end{pmatrix}
\]

By copy-and-permutation operation, we obtain a parity-check matrix of SC-LDPC code.

**Example 8.1:** This example gives an SC-LDPC code constructed from the \((3, 6)\)-regular un-coupled LDPC code. The base matrix of \((3, 6)\)-regular un-coupled LDPC code is given by \( (3 \ 3) \). Set \( B_2 = B_3 = (1 \ 1) \). Then, we obtain the base matrix of the SC-LDPC code as follows:

\[
\begin{pmatrix}
1 & 1 & O \\
1 & 1 & 1 \\
1 & 1 & 1 \\
O & \cdots
\end{pmatrix}
\]

Now, let us construct the protograph of the SC-LDPC code. Firstly, we make protograph of \((3, 6)\)-regular LDPC code, i.e., the code with the base matrix \( (3 \ 3) \). Secondly, we make \((2L + 1)\) copies of the base graph as shown in Fig. 7. In Fig. 7, each label below the base graphs represents its position. Thirdly, spreading edges of variable nodes to the neighboring check nodes as in Fig. 8, i.e., the variable nodes in the \( i \)-th position adjacent to the check nodes in the \((i-1)\)-th, \( i \)-th, and \((i + 1)\)-th positions. The resulting graph is the base graph of the SC-LDPC code. Finally, we obtain a Tanner graph of the SC-LDPC code by the copy-and-permutation operation to the base graph.

From Fig. 8, we see that there exist check nodes with small degrees in both ends of the base graph of the SC-LDPC code. Such check nodes degrade the code rate of the SC-LDPC code, i.e., an SC-LDPC code has a lower code rate than the un-coupled LDPC code. However, this loss is negligible if \( L \) is large enough.

The check nodes with small degrees provide reliable messages in the BP decoding algorithm. Hence, BP decoding propagates reliable messages from the both ends of the Tanner graph for the SC-LDPC code. As BP decoding proceeds, reliable messages reach the middle of the Tanner graph and recover the transmitted word. In other words, at the start of BP decoding, the messages in the middle of the Tanner graph may go to waste.

By using this property, the windowed decoding algorithm [78] is proposed to reduce the decoding complexity. In this algorithm, the messages are calculated only at the nodes in a range of positions, called decoding window. At the start of decoding, the decoding window is set in the left end of the Tanner graph of the SC-LDPC code. As the decoding proceeds, the decoding window moves to the right-hand side of the Tanner graph.

Kudekar et al. [79] proved that the BP decoding threshold of the SC-LDPC code equals the MAP decoding threshold of the un-coupled LDPC code. This phenomenon is called threshold saturation. In particular, if the un-coupled LDPC code achieves the channel capacity under MAP decoding, the corresponding SC-LDPC code also achieves it under BP decoding. Summarizing above, well-designed SC-LDPC codes have good decoding performance in very large code length.

9. Non-Binary LDPC Codes

Each symbol in the finite field \( \mathbb{F}_{2^m} \) of order \( 2^m \) is represented by \( m \) bits. Hence, for a code over \( \mathbb{F}_{2^m} \), the codewords of
symbol length \( n/m \) have \( n \) bit code length.

In this section, we focus on the non-binary LDPC (NB-LDPC) codes defined over \( \mathbb{F}_q \), where \( q = 2^m > 2 \). A fundamental decoding algorithm for the NB-LDPC codes over \( \mathbb{F}_q \) is a \( q \)-ary SP algorithm [80], a generalization of the SP algorithm given in Sect. 3.1. Davey and MacKay [80] reported that an NB-LDPC code outperforms binary one by comparing the decoding error rates for the regular LDPC codes over \( \mathbb{F}_2, \mathbb{F}_4 \), and \( \mathbb{F}_8 \) with the same bit code length. Hu et al. [53] optimized the degree distribution pair for the irregular binary and non-binary LDPC codes. As a result, they showed that \((2, d_c)\)-regular LDPC code has the best performance for \( q \geq 64 \) and the optimized NB-LDPC codes have lower decoding error rates than binary one for short bit code length \( n = 1008 \). Coşkun et al. [81] reported that NB-LDPC code has a lower decoding error rate than the polar code with CRC-assisted list decoding for \( n \geq 1024 \). Based on these observations, promising applications of NB-LDPC codes include communication systems with limited code length \( n \geq 1000 \).

In the \( q \)-ary SP algorithm [80], each message is represented in a vector of length \( q \). Due to the length of the message vector, the \( q \)-ary SP algorithm requires large space complexity. Moreover, the \( q \)-ary SP algorithm requires large time complexity in the check node calculation. Hence, many low-complexity decoding algorithms are proposed for the NB-LDPC codes, e.g., extended min-sum algorithm [82], bit-reliability based majority logic decoding algorithm [83], and multiple-votes symbol-flipping algorithm [84]. Nowadays, deep-learning based decoding algorithm [85] is investigated.

For more information of the NB-LDPC codes, see [86], [87].

10. Practical Use in Communication Systems

This section briefly introduces LDPC codes used in some communication system standards.

The IEEE802.11 (wifi) [5] adopts four different QC-LDPC codes with rate \( 1/2, 5/8, 3/4 \) and \( 13/16 \). All QC-LDPC codes are of length \( 672 \) and use a circulant matrix of size 42.

The IEEE802.16 (WiMAX) [6] employs QC-LDPC code with rates \( 1/2, 2/3, 3/4 \) and \( 5/6 \). In each rate, we can choose a pair of code length and circulant matrix size from (96i, 4i) for \( i = 6, 7, 8, \ldots, 24 \). Hence, the maximum code length is 2304.

The 5G NR [13] adopts two type QC-LDPC codes, called base graph 1 and base graph 2, for the error-correcting codes in the data channels. The base graph 1 (resp. 2) is defined by \( 46 \times 68 \) (resp. \( 42 \times 52 \)) base matrix. The circulant matrix size is adjusted by the payload length. The maximum code length is \( 8448 \) (resp. \( 3840 \)) for the base graph 1 (resp. 2). The rate is adjusted by selecting untransmitted bits. For more information about the LDPC codes adopted to 5G NR, see [88].

Concatenation of long BCH and LDPC codes is employed in recent broadcasting systems [7]–[11]. For example, LDPC (IRA) codes of lengths 16200 and 64800 with eleven code rates ranging from 1/4 to 9/10 are supported based on base matrix of size \( N = 360 \) in DVB-S2 [8]. A BCH code capable of correcting up to \( r \) bit errors is used for each set of code length and rate, and \( r \) is 12 in most cases. For rate-1/2 LDPC code of length 64800, input and output of BCH encoder is 32208 and 32400 bits long, respectively, and the overall code rate of the concatenated code is 0.497.

11. Conclusions

This survey has given the definition, decoding algorithm, encoding algorithm, and construction technique of LDPC codes. We also have introduced QC-LDPC code, which has been adopted by various communication standards. Moreover, we have explained several high performance LDPC codes: protograph LDPC codes, SC-LDPC codes, and NB-LDPC codes.

References

short blocks: A survey,” European Conf. on Networks and Communications (EuCNC), 2016.


[61] Y. Ikeda and T. Nozaki, “Encoding algorithm of binary and non-binary irregular LDPC codes via block triangular matrices with low weight diagonal submatrices,” 2020 International Symposium on In-


