

Chapter 3

BINARY ERASURE CHANNEL

The *binary erasure channel* (BEC) is perhaps the simplest non-trivial channel model imaginable. It was introduced by Elias as a toy example in 1954. The emergence of the internet promoted the erasure channel into the class of “real world” channels. Indeed, erasure channels can be used to model data networks, where packets either arrive correctly or are lost due to buffer overflows or excessive delays. (It is somewhat unrealistic, however, to assume that packet losses are independent.)

A priori, one might well doubt that studying the BEC will significantly advance our understanding of the general case. Quite surprisingly, however, most properties and statements that we encounter in our investigation of the BEC hold in much greater generality. Thus, the effort invested in fully understanding the BEC case will reap substantial dividends later on.

This is a long chapter. Fortunately you do not need to read it all in order to know what iterative decoding for the BEC is about. The core of the material is contained in Sections 3.1-3.14 as well as 3.24. The remaining material concerns either more specialized or less accessible topics. They can be read in almost any order.

§3.1. CHANNEL MODEL

Erasure channels model situations where information may be lost but is never corrupted. The BEC captures erasure in the simplest form: single *bits* are transmitted and either received correctly or known to be lost. The decoding problem is to find the values of the bits given the locations of the erasures and the non-erased part of the codeword. Figure 3.1 depicts the $\text{BEC}(\epsilon)$. Time, indexed by t , is discrete and the

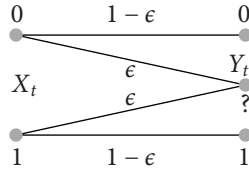


Figure 3.1: Binary erasure channel.

transmitter and receiver are synchronized (they both know t). The channel input at time t , denoted by X_t , is binary: $X_t \in \{0, 1\}$. The corresponding output Y_t takes on values in the alphabet $\{0, 1, ?\}$, where ? indicates an *erasure*. Each transmitted bit is either erased with probability ϵ , or received correctly: $Y_t \in \{X_t, ?\}$ and $P\{Y_t = ?\} = \epsilon$.

Erasure occurs for each t independently. For this reason we say that the channel is *memoryless*. The capacity of the BEC(ϵ) is $C_{\text{BEC}}(\epsilon) = 1 - \epsilon$ bits per channel use. It is easy to see that $C_{\text{BEC}}(\epsilon) \leq 1 - \epsilon$: if n bits are transmitted then, on average, $(1 - \epsilon)n$ bits are received (and they are received *correctly*). By the law of large numbers, for large n the actual number of (correctly) received bits is, with high probability, close to this average. Thus, even if the transmitter and the receiver knew in advance which bits will be erased, information can be transmitted reliably at a rate of at most $1 - \epsilon$ bits per channel use. Perhaps surprisingly, reliable transmission at a rate arbitrarily close to $1 - \epsilon$ is possible. This is confirmed in Example 3.6.

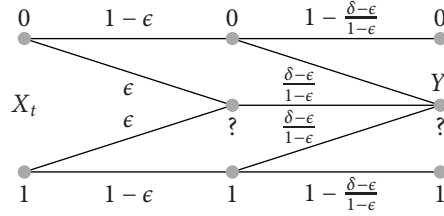


Figure 3.2: For $\epsilon \leq \delta$, the BEC(δ) is degraded with respect to the BEC(ϵ).

Consider the channel *family* $\{\text{BEC}(\epsilon)\}_{\epsilon=0}^1$. This family is *ordered* in the following sense: given two BECs, let's say with parameter ϵ and δ , $\epsilon < \delta$, we can represent the BEC(δ) as the *concatenation* of the BEC(ϵ) with a memoryless ternary-input channel as shown in Figure 3.2. Hence, the output of the BEC(δ) can be interpreted as a *degraded* version, i.e., a further perturbed version, of the output of the BEC(ϵ). We say, because this interpretation is possible, that the family $\{\text{BEC}(\epsilon)\}_{\epsilon=0}^1$ is ordered by *degradation*. This notion of degradation plays an important role in the analysis of message-passing coding systems.

§3.2. TRANSMISSION VIA LINEAR CODES

Consider a binary linear code $C[n, k]$ defined in terms of a parity-check matrix H . Assume that the transmitter chooses the codeword X uniformly at random from C and that transmission takes place over the BEC(ϵ). Let Y be the received word with elements in the extended alphabet $\{0, 1, ?\}$, where ? indicates an erasure. Let \mathcal{E} , $\mathcal{E} \subseteq [n] \triangleq \{1, \dots, n\}$, denote the *index set of erasures* and let $\bar{\mathcal{E}} \triangleq [n] \setminus \mathcal{E}$. More precisely, $i \in \mathcal{E}$ if and only if $Y_i = ?$, i.e., if the channel erased the i -th bit. Recall from page 30 that $H_{\mathcal{E}}$ denotes the submatrix of H indexed by the elements of \mathcal{E} and that $X_{\mathcal{E}}$ denotes the corresponding subvector.

§3.2.1. MAXIMUM A POSTERIORI BLOCK DECODING

Consider a MAP *block* decoder, i.e., the decoding rule is

$$(3.3) \quad \hat{x}^{\text{MAP}}(y) \triangleq \operatorname{argmax}_{x \in C} p_{X|Y}(x|y).$$

Write the defining equation $Hx^T = 0^T$ in the form $H_{\mathcal{E}}x_{\mathcal{E}}^T + H_{\bar{\mathcal{E}}}x_{\bar{\mathcal{E}}}^T = 0^T$, which, since we are working over \mathbb{F}_2 , is equivalent to

$$(3.4) \quad H_{\mathcal{E}}x_{\mathcal{E}}^T = H_{\bar{\mathcal{E}}}x_{\bar{\mathcal{E}}}^T.$$

Note that $s^T \triangleq H_{\bar{\mathcal{E}}}x_{\bar{\mathcal{E}}}^T$, the right hand side of (3.4), is *known* to the receiver since $x_{\bar{\mathcal{E}}} = y_{\bar{\mathcal{E}}}$. Consider the equation $H_{\mathcal{E}}x_{\mathcal{E}}^T = s^T$. Since, by assumption, the transmitted word is a valid codeword, we know that this equation has *at least one* solution. In formulae, $\operatorname{rank}(H_{\mathcal{E}}) \leq |\mathcal{E}|$. If $\operatorname{rank}(H_{\mathcal{E}}) = |\mathcal{E}|$, then MAP block decoding can be accomplished by solving $H_{\mathcal{E}}x_{\mathcal{E}}^T = s^T$. On the other hand, there are *multiple* solutions (i.e., the MAP decoder is not able to recover the codeword uniquely) if and only if $\operatorname{rank}(H_{\mathcal{E}}) < |\mathcal{E}|$. More formally, let

$$\mathcal{X}^{\text{MAP}}(y) \triangleq \{x \in C : H_{\mathcal{E}}x_{\mathcal{E}}^T = H_{\bar{\mathcal{E}}}y_{\bar{\mathcal{E}}}^T; x_{\bar{\mathcal{E}}} = y_{\bar{\mathcal{E}}}\},$$

i.e., $\mathcal{X}^{\text{MAP}}(y)$ is the set of all codewords *compatible* with the received word y . Since the prior is uniform, (3.3) becomes

$$\hat{x}^{\text{MAP}}(y) = \operatorname{argmax}_{x \in C} p_{Y|X}(y|x).$$

Now, for any codeword x , if $x_{\bar{\mathcal{E}}} \neq y_{\bar{\mathcal{E}}}$, then $p_{Y|X}(y|x) = 0$ and if $x_{\bar{\mathcal{E}}} = y_{\bar{\mathcal{E}}}$, then $p_{Y|X}(y|x) = (1 - \epsilon)^{n-|\mathcal{E}|} \epsilon^{|\mathcal{E}|}$. Thus, all elements of $\mathcal{X}^{\text{MAP}}(y)$ are equally likely and the transmitted vector x is either uniquely determined by y or there are multiple solutions. Therefore we say,

$$\hat{x}^{\text{MAP}}(y) = \begin{cases} x \in \mathcal{X}^{\text{MAP}}(y), & \text{if } |\mathcal{X}^{\text{MAP}}(y)| = 1, \\ ?, & \text{otherwise.} \end{cases}$$

We remark that a ? (erasure) is not the same as an *error*. The correct solution x is an element of $\mathcal{X}^{\text{MAP}}(y)$.

§3.2.2. MAXIMUM A POSTERIORI BIT DECODING

Consider now the MAP *bit* decoder that uses the decoding rule

$$(3.5) \quad \hat{x}_i^{\text{MAP}}(y) \triangleq \operatorname{argmax}_{\alpha \in \{0,1\}} p_{X_i|Y}(\alpha|y).$$

If $i \in \mathcal{E}$, when can x_i be recovered? Intuitively, we expect that x_i can be recovered if and only if all elements of $\mathcal{X}^{\text{MAP}}(y)$ have the same value for the i -th bit. This is in fact

correct. More specifically, we claim that x_i can *not* be recovered if and only if $H_{\{i\}}$ is an element of the space spanned by the columns of $H_{\mathcal{E} \setminus \{i\}}$. This is equivalent to the statement that $Hw^T = 0^T$ has a solution with $w_i = 1$ and $w_{\bar{\mathcal{E}}} = 0$. Now, if there is such a solution then for every element x of $\mathcal{X}^{\text{MAP}}(y)$ we also have $x + w \in \mathcal{X}^{\text{MAP}}(y)$. It follows that exactly half the elements $x \in \mathcal{X}^{\text{MAP}}(y)$ have $x_i = 0$ and half have $x_i = 1$. Conversely, if we can find two elements x and x' of $\mathcal{X}^{\text{MAP}}(y)$ with $x_i \neq x'_i$, then $w = x + x'$ solves $Hw^T = 0^T$ and has $w_i = 1$ and $w_{\bar{\mathcal{E}}} = 0$. Proceeding formally, we get

$$\begin{aligned} \hat{x}_i^{\text{MAP}}(y) &= \operatorname{argmax}_{\alpha \in \{0,1\}} p_{X_i|Y}(\alpha|y) = \operatorname{argmax}_{\alpha \in \{0,1\}} \sum_{x \in \{0,1\}^n: x_i = \alpha} p_{X|Y}(x|y) \\ &= \operatorname{argmax}_{\alpha \in \{0,1\}} \sum_{x \in C: x_i = \alpha} p_{X|Y}(x|y) = \begin{cases} \alpha, & \text{if } \forall x \in \mathcal{X}^{\text{MAP}}(y), x_i = \alpha, \\ ?, & \text{otherwise.} \end{cases} \end{aligned}$$

We conclude that optimal (block or bit) decoding for the BEC(ϵ) can be accomplished in complexity at most $O(n^3)$ by solving a linear system of equations (e.g., by Gaussian elimination). Further, we have a characterization of decoding failures of a MAP decoder for both the block and the bit erasure case in terms of rank conditions.

EXAMPLE 3.6 (PERFORMANCE OF $\mathcal{H}(n, k)$). In Problem 3.21 you are asked to show that the average block erasure probability of Gallager's parity-check ensemble (see Definition 1.26) satisfies

$$\begin{aligned} \mathbb{E}_{\mathcal{H}(n,k)}[P_B(H, \epsilon)] &\leq \sum_{e=0}^{n-k} \binom{n}{e} \epsilon^e (\bar{\epsilon})^{n-e} 2^{e-n+k} + \sum_{e=n-k+1}^n \binom{n}{e} \epsilon^e (\bar{\epsilon})^{n-e} \\ &= 2^{k-n} (\bar{\epsilon})^n \sum_{e=0}^{n-k} \binom{n}{e} \left(\frac{2\epsilon}{\bar{\epsilon}}\right)^e + (\bar{\epsilon})^n \sum_{e=n-k+1}^n \binom{n}{e} \left(\frac{\epsilon}{\bar{\epsilon}}\right)^e, \end{aligned}$$

where the bound is loose by a factor of at most two. Consider the asymptotic case as $n \rightarrow \infty$. Assume that $r = k/n = (1 - \delta)C_{\text{BEC}}(\epsilon)$, where $\frac{1}{1+\epsilon} < 1 - \delta < 1$. The elements of both sums are *unimodal* sequences, see Appendix D, i.e., they are first increasing up to their maximum value, after which they decrease. Consider the terms $\left\{ \binom{n}{e} \left(\frac{2\epsilon}{\bar{\epsilon}}\right)^e \right\}_{e=0}^{n-k}$ of the first sum. Because of our assumption $\frac{1}{1+\epsilon} < 1 - \delta$, the upper summation index $n(1 - r)$ is to the left of the maximum, which occurs at $e \approx \frac{2\epsilon}{1+\epsilon}n$. The first sum can therefore be upper bounded by the last term times the number of summands. Similarly, the maximum term of $\left\{ \binom{n}{e} \left(\frac{\epsilon}{\bar{\epsilon}}\right)^e \right\}_{e=n-k+1}^n$ occurs at $e \approx \epsilon n$, which, since $1 - \delta < 1$, is to the left of the lower summation index $e = n - k + 1$. This second sum can therefore be upper bounded by the first term times the number of summands. This leads to the bound

$$(3.7) \quad \mathbb{E}_{\mathcal{H}(n,rn)}[P_B^{\text{MAP}}(H, \epsilon)] \leq (n+1)2^{-nD_2(\epsilon(1+\delta\frac{\epsilon}{\bar{\epsilon}}))\|\epsilon\|},$$

where we defined $D_2(\alpha \parallel \beta) \triangleq -\alpha \log_2 \frac{\beta}{\alpha} - \bar{\alpha} \log_2 \frac{\bar{\beta}}{\bar{\alpha}}$ and used the bound $\binom{n}{m} \leq \sum_{k=0}^m \binom{n}{k} \stackrel{(1.59)}{\leq} 2^{nh_2(m/n)}$, $m \leq n/2$. The quantity $D_2(\cdot, \cdot)$ is known as the *Kullback-Leibler* distance (between two Bernoulli distributions with parameters α and β , respectively). Let $\alpha, \beta \in (0, 1)$. Using Jensen's inequality (1.61), we conclude that

$$D_2(\alpha \parallel \beta) = -\alpha \log_2 \frac{\beta}{\alpha} - \bar{\alpha} \log_2 \frac{\bar{\beta}}{\bar{\alpha}} \geq -\log_2 \left(\alpha \frac{\beta}{\alpha} + \bar{\alpha} \frac{\bar{\beta}}{\bar{\alpha}} \right) = 0,$$

with strict inequality if $\alpha \neq \beta$. Therefore, for $\delta \in (0, 1]$ the right hand side of (3.7) tends to zero exponentially fast in the blocklength n . We conclude that reliable transmission at any rate $r = (1 - \delta)C_{\text{BEC}}(\epsilon)$, $\delta > 0$, is possible. In words, reliable transmission up to $C_{\text{BEC}}(\epsilon)$ is possible, as promised. \diamond

§3.3. TANNER GRAPHS

Let C be binary linear code and let H be a parity-check matrix of C , i.e., $C = C(H)$. Recall that, by our convention, we do not require the rows of H to be linearly independent. Assume that H has dimensions $m \times n$. In Example 2.4 on page 48 we introduced the Tanner graph associated with a code C . This is the graph which visualizes the factorization of the code membership function. Since this graph plays a central role let us repeat its definition.

The Tanner graph associated with H is a *bipartite* graph. It has n *variable* nodes, corresponding to the components of the codeword, and m *check* nodes, corresponding to the set of parity-check constraints (rows of H). Check node j is connected to variable node i if $H_{ji} = 1$, i.e., if variable i participates in the j -th parity-check constraint. Since there are many parity-check matrices representing the same code, there are many Tanner graphs corresponding to a given C . Although all of these Tanner graphs describe the same code, they are not equivalent from the point of view of the message-passing decoder (see Problem 3.15.)

EXAMPLE 3.8 ((3,6)-REGULAR CODE). Consider the parity-check matrix

$$(3.9) \quad \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{matrix} & \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{pmatrix} \triangleq H.$$

The bipartite graph representing $C(H)$ is shown on the left of Figure 3.10. Each check

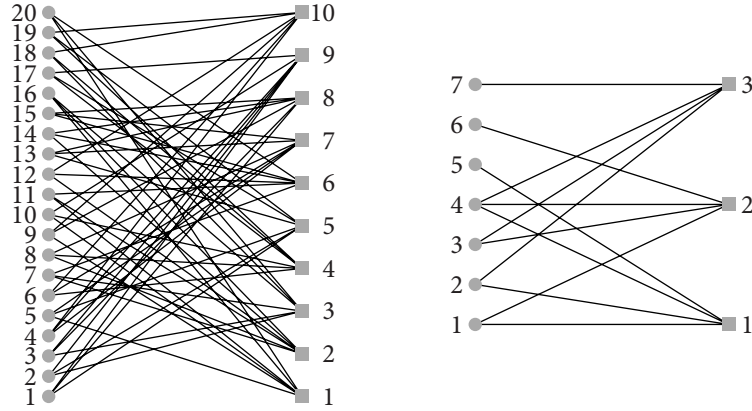


Figure 3.10: Left: Tanner graph of H given in (3.9). Right: Tanner graph of $[7, 4, 3]$ Hamming code corresponding to the parity-check matrix on page 15. This graph is discussed in Example 3.11.

node represents one linear constraint (one row of H). For the particular example we start with twenty degrees of freedom (twenty variable nodes). The ten constraints reduce the number of degrees of freedom by at most ten (and exactly by ten if all these constraints are linearly independent as in this specific example). Therefore at least ten degrees of freedom remain. It follows that the shown code has rate (at least) one-half. \diamond

§3.4. LOW-DENSITY PARITY-CHECK CODES

We are ready to introduce the class of codes at the heart of this book. In a nutshell, *low-density parity-check* (LDPC) codes are linear codes which have at least one *sparse* Tanner graph. The primary reason for focussing at such codes is that, as we will see shortly, they exhibit good performance under message-passing decoding.

Consider again the Tanner graph on the left of Figure 3.10. Each variable node has degree 3 and every check node has degree 6. We call such a code a $(3, 6)$ -regular LDPC code. More generally, an $(1, r)$ -regular LDPC code is a binary linear code such that every variable node has degree 1 and every check node has degree r . Why *low-density*? The number of edges in the Tanner graph of a $(1, r)$ -regular LDPC code is $1n$, where n is the length of the code. As n increases, the number edges in the Tanner graph grows *linearly* in n . This is in contrast to codes in the parity-check ensemble $\mathcal{H}(n, nr)$ where the number of edges in their Tanner graph grows like the *square* of the code length n .

The behavior of LDPC codes can be significantly improved by allowing nodes of different degrees as well as other structural improvements. We define an *irregular* LDPC code as an LDPC code for which the degrees of nodes are chosen according to some distribution.

EXAMPLE 3.11 (TANNER GRAPH OF $[7, 4, 3]$ HAMMING CODE). The right hand side of Figure 3.10 shows the Tanner graph of the $[7, 4, 3]$ Hamming code corresponding to the parity-check matrix on page 15. The three check nodes have degree 4. There are three variable nodes of degree 1, three variable nodes of degree 2, and one variable node of degree 3. \diamond

Assume that the LDPC code has length n and that the number of variable nodes of degree i is Λ_i , so that $\sum_i \Lambda_i = n$. In the same fashion, denote the number of check nodes of degree i by P_i , so that $\sum_i P_i = n\bar{r}$, where r is the *design* rate (ratio of length minus number of constraints and the length) of the code and \bar{r} is a shorthand for $1 - r$. Further, since the edge counts must match up, $\sum_i i\Lambda_i = \sum_i iP_i$. It is convenient to introduce the following compact notation,

$$(3.12) \quad \Lambda(x) \triangleq \sum_{i=1}^{l_{\max}} \Lambda_i x^i, \quad P(x) \triangleq \sum_{i=1}^{r_{\max}} P_i x^i,$$

i.e., $\Lambda(x)$ and $P(x)$ are polynomials with non-negative expansions around zero whose integral coefficients are equal to the number of nodes of various degrees. From these definitions we see immediately the following relationships

$$(3.13) \quad \Lambda(1) = n, \quad P(1) = n\bar{r}, \quad r(\Lambda, P) = 1 - \frac{P(1)}{\Lambda(1)}, \quad \Lambda'(1) = P'(1).$$

We call Λ and P the variable and check *degree distributions from a node perspective*. Sometimes it is more convenient to use the *normalized* degree distributions

$$L(x) \triangleq \frac{\Lambda(x)}{\Lambda(1)}, \quad R(x) \triangleq \frac{P(x)}{P(1)}.$$

instead.

EXAMPLE 3.14 (DEGREE DISTRIBUTION OF $[7, 4, 3]$ HAMMING CODE). We have

$$\begin{aligned} \Lambda(x) &= 3x + 3x^2 + x^3, & P(x) &= 3x^4, \\ L(x) &= \frac{3}{7}x + \frac{3}{7}x^2 + \frac{1}{7}x^3, & R(x) &= x^4. \end{aligned} \quad \diamond$$

DEFINITION 3.15 (THE STANDARD ENSEMBLE LDPC (Λ, P)). Given a degree distribution pair (Λ, P) , define an *ensemble* of bipartite graphs LDPC (Λ, P) in the following way. Each graph in LDPC (Λ, P) has $\Lambda(1)$ variable nodes and $P(1)$ check nodes: Λ_i variable nodes and P_i check nodes have degree i . A node of degree i has i *sockets* from which the i edges emanate, so that in total there are $\Lambda'(1) = P'(1)$ sockets on each side. Label the sockets on each side with the set $[\Lambda'(1)] \triangleq \{1, \dots, \Lambda'(1)\}$ in some arbitrary but fixed way. Let σ be a permutation on $[\Lambda'(1)]$. Associate to σ a bipartite graph by connecting the i -th socket on the variable side to the $\sigma(i)$ -th socket on the check side. Letting σ run over the set of permutations on $[\Lambda'(1)]$ generates a set of bipartite graphs. Finally, we define a probability distribution over the set of graphs by placing the uniform probability distribution on the set of permutations. This is the ensemble of bipartite graphs LDPC (Λ, P) . It remains to associate a code with every element of LDPC (Λ, P) . We will do so by associating a parity-check matrix to each graph. Because of possible multiple edges and since the encoding is done over the field \mathbb{F}_2 , we define the parity-check matrix H as that $\{0, 1\}$ -matrix which has a non-zero entry at row i and column j if and only if the i -th check node is connected to the j -th variable node an *odd* number of times.

Since for every graph there is an associated code we use these two terms interchangeably and we refer, e.g., to codes as elements of LDPC (Λ, P) .

This is a somewhat subtle point: graphs are *labeled* (more precise, they have labeled sockets) and have a uniform probability distribution; the induced codes are unlabeled and the probability distribution on the set of codes is not necessarily the uniform one. Therefore, if in the sequel we say that we pick a code uniformly at random we really mean that we pick a graph at random from the ensemble of graphs and consider the induced code. This convention should not cause any confusion and simplifies our notation considerably. ∇

As discussed in Problem 3.7, ensembles with a positive fraction of degree-one variable nodes have non-zero bit error probability for *all* non-zero channel parameters even in the limit of infinite blocklengths. The reason for this error floor is that by our definition of the ensemble (where variable nodes are matched randomly to check nodes) there is a positive probability that two degree-one variable nodes connect to the same check node and such a code contains codewords of weight two. Therefore, we typically only consider ensembles without degree-one nodes. But, as we will discuss in Chapter 7, it is possible to introduce degree-one variable nodes if their edges are placed with care.

For the asymptotic analysis it is more convenient to take on an *edge perspective*. Define

$$(3.16) \quad \lambda(x) = \sum_i \lambda_i x^{i-1} \triangleq \frac{\Lambda'(x)}{\Lambda'(1)} = \frac{L'(x)}{L'(1)}, \quad \rho(x) = \sum_i \rho_i x^{i-1} \triangleq \frac{P'(x)}{P'(1)} = \frac{R'(x)}{R'(1)}.$$

Note that λ and ρ are polynomials with non-negative expansions around zero. Some thought shows that λ_i (ρ_i) is equal to the *fraction of edges* that connect to variable (check) nodes of degree i , see Problem 3.3. In other words, λ_i (ρ_i) is the probability that an edge chosen uniformly at random from the graph is connected to a variable (check) node of degree i . We call λ and ρ the variable and check *degree distributions from an edge perspective*. The inverse relationships read

$$(3.17) \quad \frac{\Lambda(x)}{n} = L(x) = \frac{\int_0^x \lambda(z) dz}{\int_0^1 \lambda(z) dz}, \quad \frac{P(x)}{n\bar{r}} = R(x) = \frac{\int_0^x \rho(z) dz}{\int_0^1 \rho(z) dz}.$$

As discussed in Problem 3.4 and 3.5, the average variable and check node degrees, call them \mathbf{l}_{avg} and \mathbf{r}_{avg} , can be expressed as

$$(3.18) \quad \mathbf{l}_{\text{avg}} = L'(1) = \frac{1}{\int_0^1 \lambda(x) dx}, \quad \mathbf{r}_{\text{avg}} = R'(1) = \frac{1}{\int_0^1 \rho(x) dx},$$

respectively, and the *design* rate is given by

$$(3.19) \quad r(\lambda, \rho) = 1 - \frac{\mathbf{l}_{\text{avg}}}{\mathbf{r}_{\text{avg}}} = 1 - \frac{L'(1)}{R'(1)} = 1 - \frac{\int_0^1 \rho(x) dx}{\int_0^1 \lambda(x) dx}.$$

The design rate is the rate of the code assuming that all constraints are linearly independent.

EXAMPLE 3.20 (CONVERSION OF DEGREE DISTRIBUTIONS: HAMMING CODE). For the $[7, 4, 3]$ Hamming code we have

$$\lambda(x) = \frac{1}{4} + \frac{1}{2}x + \frac{1}{4}x^2, \quad \rho(x) = x^3. \quad \diamond$$

EXAMPLE 3.21 (CONVERSION OF DEGREE DISTRIBUTIONS: SECOND EXAMPLE). Consider the pair (Λ, P)

$$\Lambda(x) \triangleq 613x^2 + 202x^3 + 57x^4 + 84x^7 + 44x^8, \quad P(x) \triangleq 500x^6,$$

with

$$\Lambda(1) = 1000, \quad P(1) = 500, \quad \Lambda'(1) = P'(1) = 3000.$$

This pair represents an ensemble of codes of length 1000 and of (design) rate one-half. Converting into edge perspective we get

$$\lambda(x) = \frac{1226}{3000}x + \frac{606}{3000}x^2 + \frac{228}{3000}x^3 + \frac{588}{3000}x^6 + \frac{352}{3000}x^7, \quad \rho(x) = x^5.$$

◇

Since (Λ, P) , (n, L, R) and (n, λ, ρ) contain equivalent information, we frequently and freely switch between these various perspectives. We write $(\Lambda, P) \triangleq (n, L, R) \triangleq (n, \lambda, \rho)$ if we want to express the fact that degree distributions in different formats are equivalent. We therefore often refer to the standard ensemble as LDPC (n, λ, ρ) . For the asymptotic analysis it is convenient to fix (λ, ρ) and to investigate the performance of the ensemble LDPC (n, λ, ρ) as the blocklength n tends to infinity. For some n the corresponding (Λ, P) is not integral. We assume in such a scenario that the individual node distributions are rounded to the closest integer. In any case, sublinear (in n) deviations of degree distributions have no effect on the asymptotic performance or rate of the ensemble. In the sequel we therefore ignore this issue.

The design rate as defined in (3.19) is in general only a lower bound on the actual rate because the parity-check matrix can have linearly dependent rows. The following lemma asserts that, under some technical conditions, the actual rate of a random element of an ensemble is equal to the design rate with high probability as the blocklength increases.

LEMMA 3.22 (RATE VERSUS DESIGN RATE). Consider the ensemble LDPC $(n, \lambda, \rho) \triangleq$ LDPC (n, L, R) . Let $r(\lambda, \rho)$ denote the design rate of the ensemble and let $r(G)$ denote the actual rate of a code G , $G \in \text{LDPC}(n, \lambda, \rho)$. Consider the function $\Psi(y)$,

$$\begin{aligned} \Psi(y) = & -L'(1) \log_2 \left[\frac{(1+yz)}{(1+z)} \right] + \sum_i L_i \log_2 \left[\frac{1+y^i}{2} \right] \\ & + \frac{L'(1)}{R'(1)} \sum_j R_j \log_2 \left[1 + \left(\frac{1-z}{1+z} \right)^j \right], \end{aligned} \quad (3.23)$$

$$(3.24) \quad z = \left(\sum_i \frac{\lambda_i y^{i-1}}{1 + y^i} \right) / \left(\sum_i \frac{\lambda_i}{1 + y^i} \right).$$

If for $y \geq 0$, $\Psi(y) \leq 0$ with equality only at $y = 1$ then

$$P\{r(\mathbb{G})n = r(\lambda, \rho)n - \nu\} = 1 - o_n(1),$$

where $\nu = 1$ if all variable nodes have even degree, and $\nu = 0$ otherwise.

Discussion: The extra constant ν is easily explained. If all variable nodes have even degree then each variable appears in an even number of constraints and so the sum of all constraints is zero. This means that there is at least one linearly dependent equation in this case. The lemma asserts that (assuming the technical condition is fulfilled) all other equations are linearly independent with high probability.

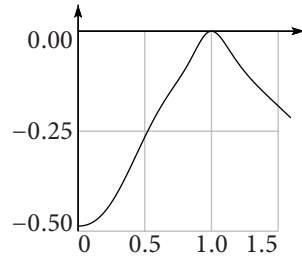


Figure 3.26: The function $\Psi(y)$.

EXAMPLE 3.25 (DEGREE DISTRIBUTION ($\lambda(x) = \frac{3x+3x^2+4x^{13}}{10}$, $\rho(x) = x^6$)). The function $\Psi(y)$ is shown in Figure 3.26. According to this plot, $\Psi(y) \leq 0$ with equality only at $y = 1$. By Lemma 3.22, the probability that the rate of a random element of this ensemble is equal to its design rate converges to one as the blocklength tends to infinity.

Discussion: The proof of Lemma 3.22 is quite technical and so we relegate it to Section 3.24. As we will discuss in more detail, the function $\Psi(y)$ represents the log of the number of codewords of a given weight divided by the length of the code, with a parametrization so that $y = 1$ corresponds to the codewords of relative weight one-half. The lemma asserts that if the “dominant” codewords in the code are those of relative weight one-half then, for sufficiently large blocklengths, the actual rate is equal to the design rate with high probability. \diamond

For *regular* ensembles the actual rate always converges to the design rate.

COROLLARY 3.27 (DESIGN RATE EQUALS RATE FOR REGULAR ENSEMBLES). Consider the regular ensemble LDPC $(nx^1, n^{\frac{1}{r}}x^r)$ with $2 \leq 1 < r$. Let $r(1, r) = 1 - \frac{1}{r}$

denote the design rate of the ensemble and let $r(G)$ denote the actual rate of a code G , $G \in \text{LDPC}(nx^1, n\frac{1}{r}x^r)$. Then

$$P\{r(G)n = r(1, r)n - v\} = 1 - o_n(1),$$

where $v = 1$ if 1 is even, and $v = 0$ otherwise.

Proof. In the regular case the expression of $\Psi(y)$ simplifies to

$$\Psi(y) = \log\left(\frac{1}{2}(1 + y^1)^{1-1}((1 + y^{1-1})^r + (1 - y^{1-1})^r)^{\frac{1}{r}}\right).$$

Define $x \triangleq y^{1-1}$. Then the condition $\Psi(y) \leq 0$, with strict inequality except for $y = 1$, is equivalent to $f(x, r) \leq g(x, 1)$, with strict inequality except for $x = 1$, where $f(x, r) \triangleq ((1 + x)^r + (1 - x)^r)^{\frac{1}{r}}$ and $g(x, 1) \triangleq 2^{\frac{1}{r}}(1 + x^{\frac{1}{1-1}})^{\frac{1-1}{r}}$. We start by showing that for $r \geq 2$ and $x \geq 0$, $f(x, r) \leq g(x, r)$. To see this, consider the equivalent statement $2 \sum_i \binom{r}{2i} x^{2i} = f(x, r)^r \leq g(x, r)^r = 2 \sum_j \binom{r-1}{j} x^{\frac{r-1}{r-1}j}$. For $r = 2$ a direct check shows that the two sides are equal and the same is true for $x = 0$. Consider therefore the case $r \geq 3$ and $x > 0$. First cancel the factor 2 from both sides. Next note that both series start with the term 1 and if r is even then the last term on both sides is x^r . For each remaining term $\binom{r}{2i} x^{2i}$, $2 \leq 2i < r$, on the left there are exactly two terms on the right of the form $\binom{r-1}{2i-1} x^{\frac{(2i-1)r}{r-1}} + \binom{r-1}{2i} x^{\frac{2ir}{r-1}}$. Now note that for $x > 0$, x^α is a convex- \cup function in α for $\alpha > 0$ and that $\binom{r-1}{2i-1} + \binom{r-1}{2i} = \binom{r}{2i}$. Therefore by Jensen's inequality (1.61),

$$\frac{\binom{r-1}{2i-1}}{\binom{r}{2i}} x^{\frac{(2i-1)r}{r-1}} + \frac{\binom{r-1}{2i}}{\binom{r}{2i}} x^{\frac{2ir}{r-1}} \geq \left(x^{\left(\binom{r-1}{2i-1} \frac{(2i-1)r}{r-1} + \binom{r-1}{2i} \frac{2ir}{r-1} \right) / \binom{r}{2i}} \right) = x^{2i}.$$

Now where we know that $f(x, r) \leq g(x, r)$ for $r \geq 2$ and $x \geq 0$, the proof will be complete if we can show that $g(x, 1)$ is a decreasing function in 1 and that it is strictly decreasing except for $x = 1$: we write $f(x, r) \leq g(x, r) \stackrel{1 \leq r}{\leq} g(x, 1)$, where the last inequality is strict for $x \neq 1$. It is the task of Problem 3.1 to show that $g(x, 1)$ is indeed decreasing in 1 . \square

An example where the technical condition is *not* fulfilled is discussed in Problem 3.37.

§3.5. MESSAGE-PASSING DECODER

In Chapter 2 we introduced a message-passing algorithm to accomplish the decoding task. Let us specialize this algorithm for the BEC.

The Tanner graph of an LDPC code (and so also the factor graph corresponding to bit-wise MAP decoding) is in general not a tree. We nevertheless use the standard message-passing rules summarized in Figure 2.11. If the factor graph is a tree there is a natural *schedule* given by starting at the leaf nodes and sending a message once all incoming messages required for the computation have arrived. But in order to completely define the algorithm for a code with cycles we need to *specify* a schedule. In general, different schedules can lead to different performance. This is our convention: we proceed in *rounds*; a round starts by processing incoming messages at check nodes and then sending the resulting outgoing messages to variable nodes along all edges. These messages are subsequently processed at the variable nodes and the outgoing messages are sent back along all edges to the check nodes. This constitutes one round of message-passing. In general, decoding consists of several such rounds. In iteration zero, there is no problem-specific information that we can send from the check nodes to the variable nodes. Therefore, in this initial round, the variable nodes simply send the messages received from the channel to their neighboring check nodes.

These initial messages $(\mu_j(0), \mu_j(1)) = (p_{Y_j|X_j}(y_j|0), p_{Y_j|X_j}(y_j|1))$ are either $(1 - \epsilon, 0)$, (ϵ, ϵ) , or $(0, 1 - \epsilon)$. This corresponds to the three possibilities, namely that the received values are 0, ? (erasure), or 1, respectively.¹ Recall that the *normalization* of the messages plays no role (we saw in Section 2.5.2 that we only need to know the ratio) so that equivalently we can work with the set of messages $(1, 0)$, $(1, 1)$, and $(0, 1)$. In the sequel we will call these also the “0” (zero), the “?” (erasure), and the “1” (one), message. Therefore, e.g., 0, $(1, 0)$, and “zero”, all refer to the same message.

We now get to the processing rules. We claim that for the BEC the general message-passing rules summarized in Figure 2.11 simplify to the following: at a variable node the outgoing message is an erasure if *all* incoming messages are erasures. Otherwise, since the channel *never* introduces errors, all non-erasure messages must agree and either be 0 or 1. In this case the outgoing message is equal to this common value. At a check node the outgoing message is an erasure if *any* of the incoming messages is an erasure. Otherwise, if all of the incoming messages are either 0 or 1 then the outgoing message is the mod-2 sum of the incoming messages.

Consider the first claim: if all messages entering a variable node are from the set $\{(1, 0), (1, 1)\}$, then the outgoing message (which is equal to the componentwise product of the incoming messages according to the general message-passing rules) is also from this set. Further, it is equal to $(1, 1)$ (i.e., an erasure) only if *all* incoming messages are of the form $(1, 1)$ (i.e., erasures). The equivalent statement is true if all incoming messages are from the set $\{(0, 1), (1, 1)\}$. (Since the channel never introduces errors we only need to consider these two cases.)

¹We assume in this chapter that the bits take on the values 0 and 1.

Next consider the claim concerning the message-passing rule at a check node: it suffices to consider a check node of degree three with two incoming messages since check nodes of higher degree can be modeled as the cascade of several check nodes, each of which has two inputs and one output (e.g., $x_1 + x_2 + x_3 = (x_1 + x_2) + x_3$). Let $(\mu_1(0), \mu_1(1))$ and $(\mu_2(0), \mu_2(1))$ denote the incoming messages. By the standard message-passing rules the outgoing message is

$$\begin{aligned} (\mu(0), \mu(1)) &= \left(\sum_{x_1, x_2} \mathbb{1}_{\{x_1+x_2=0\}} \mu_1(x_1) \mu_2(x_2), \sum_{x_1, x_2} \mathbb{1}_{\{x_1+x_2=1\}} \mu_1(x_1) \mu_2(x_2) \right) \\ &= (\mu_1(0)\mu_2(0) + \mu_1(1)\mu_2(1), \mu_1(0)\mu_2(1) + \mu_1(1)\mu_2(0)). \end{aligned}$$

If $(\mu_2(0), \mu_2(1)) = (1, 1)$ then, up to normalization, $(\mu(0), \mu(1)) = (1, 1)$. This shows that if any of the inputs is an erasure then the output is an erasure. Further, if both messages are known, i.e., if both are from the set $\{0, 1\}$, then an explicit check shows that the message-passing rules correspond to the mod-2 sum.

Figure 3.28 shows the application of the message-passing decoder to the $[7, 4, 3]$ Hamming code assuming that the received word is $(0, ?, ?, 1, 0, ?, 0)$. In iteration 0 the variable-to-check messages correspond to the received values. Consider the check-to-variable message sent in iteration 1 from check node 1 to variable node 2 (this is shown on the left of Figure 3.28 in the second from the top figure). This message is 1 (the mod-2 sum of incoming messages) according to the message-passing rule. This is quite intuitive: this message reflects the fact that through the parity-check constraint $x_1 + x_2 + x_4 + x_5 = 0$ we can find x_2 given x_1, x_4 , and x_5 . Although this might not be completely obvious at this point, this message-passing algorithm is entirely equivalent to the greedy algorithm based on the Venn diagram description which we discussed in Section 1.9. In other words: for the BEC message-passing is equivalent to greedily checking whether any of the parity-constraints allows us to find an yet unknown value from already known ones. After three iterations the transmitted word is found to be $(0, 1, 0, 1, 0, 1, 0)$.

§3.6. TWO BASIC SIMPLIFICATIONS

In the previous sections we have introduced code ensembles and a low-complexity message-passing decoder. We start our investigation of how well this combination performs.

§3.6.1. RESTRICTION TO THE ALL-ZERO CODEWORD

The first big simplification stems from the realization that the performance is *independent* of the transmitted codeword and is only a function of the erasure pattern: at any iteration the set of known variable nodes is only a function of the *set* of known messages but independent of their values. The equivalent statement is true for the set of known check nodes.

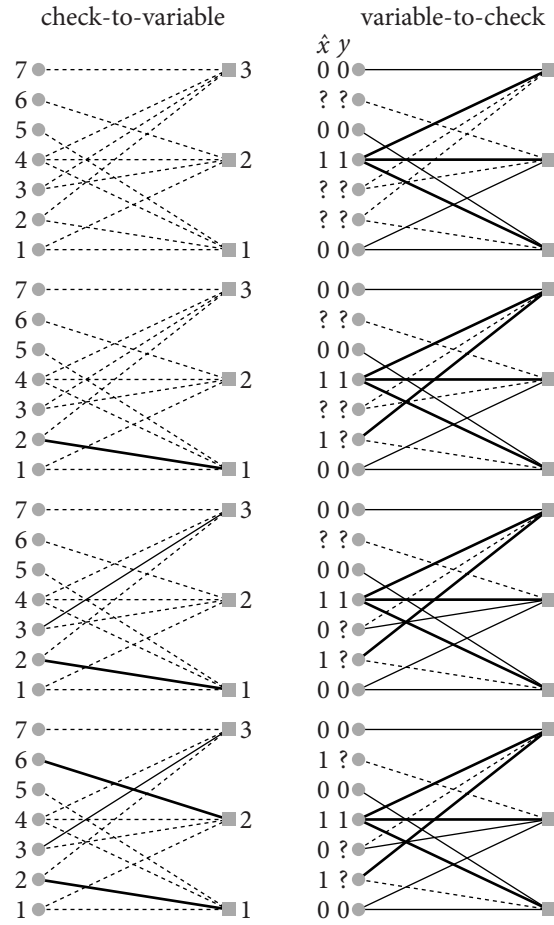


Figure 3.28: Message-passing decoding of the $[7, 4, 3]$ Hamming code with the received word $y = (0, ?, ?, 1, 0, ?, 0)$. The vector \hat{x} denotes the current estimate of the transmitted word x . A 0 message is indicated as thin line, a 1 message is indicated as thick line, and a ? message is drawn as dashed line. The four rows correspond to iterations 0 to 3. After the first iteration we recover $x_2 = 1$, after the second $x_3 = 0$, and after the third we know that $x_6 = 1$. This means that for this case we can recover the codeword and it is $x = (0, 1, 0, 1, 0, 1, 0)$.

FACT 3.29 (CONDITIONAL INDEPENDENCE OF ERASURE PROBABILITY). Let G be the Tanner graph representing a binary linear code C . Assume that C is used to transmit over the BEC(ϵ) and assume that the decoder performs message-passing decoding on G . Let $P^{\text{BP}}(G, \epsilon, \ell, x)$ denote the conditional (bit or block) probability of erasure after the ℓ -th decoding iteration, assuming that x was sent, $x \in C$. Then $P^{\text{BP}}(G, \epsilon, \ell, x) = \frac{1}{|C|} \sum_{x' \in C} P^{\text{BP}}(G, \epsilon, \ell, x') \triangleq P^{\text{BP}}(G, \epsilon, \ell)$, i.e., $P^{\text{BP}}(G, \epsilon, \ell, x)$ is independent of the transmitted codeword.

As a consequence, we are free to choose a particular codeword and to analyze the performance of the system assuming that this codeword was sent. It is natural to assume that the all-zero word, which is contained in every linear code, was sent. We refer to this assumption as the “all-zero codeword” assumption.

A word about notation: *iterative* decoding is a generic term referring to decoding algorithms which proceed in iterations. A sub-class of iterative algorithms are *message-passing* algorithms (like the algorithm which we introduced in the previous section). Message-passing algorithms are iterative algorithms which obey the *message-passing paradigm*: this means that an outgoing message along an edge e only depends on the incoming messages along all edges *other* than e itself. The message-passing algorithm which we introduced in Chapter 2 and which we adopted in this chapter is a special case in which the messages represent probabilities or “beliefs.” The algorithm is therefore also known as *belief propagation* (BP) algorithm. For the BEC essentially any meaningful message-passing algorithm is equivalent to the BP algorithm. But for the general case message-passing algorithms other than the BP algorithm play an important role. For the remainder of this chapter we use the shorthand BP to refer to the decoder.

§3.6.2. CONCENTRATION

Rather than analyzing individual codes it suffices to assess the *ensemble average performance*. This is true, since, as the next theorem asserts, the individual elements of an ensemble behave with high probability close to the ensemble average.

THEOREM 3.30 (CONCENTRATION AROUND ENSEMBLE AVERAGE). Let G , chosen uniformly at random from LDPC(n, λ, ρ), be used for transmission over the BEC(ϵ). Assume that the decoder performs ℓ rounds of message-passing decoding and let $P_b^{\text{BP}}(G, \epsilon, \ell)$ denote the resulting bit erasure probability. Then, for ℓ fixed and for any given $\delta > 0$, there exists an $\alpha > 0$, $\alpha = \alpha(\lambda, \rho, \epsilon, \delta, \ell)$, such that

$$P\{|P_b^{\text{BP}}(G, \epsilon, \ell) - \mathbb{E}_{G' \in \text{LDPC}(n, \lambda, \rho)}[P_b^{\text{BP}}(G', \epsilon, \ell)]| > \delta\} \leq e^{-\alpha n}.$$

In words, the theorem asserts that all except an exponentially (in the block-length) small fraction of codes behave within an arbitrarily small δ from the ensemble average. Assuming sufficiently large blocklengths, the ensemble average is

a good indicator for the individual behavior. We therefore focus our effort on the design and construction of ensembles whose average performance approaches the Shannon theoretic limit.

EXAMPLE 3.31 (CONCENTRATION FOR LDPC ($\Lambda(x) = 512x^3, P(x) = 256x^6$)). Figure 3.32 shows the erasure probability curves under BP decoding for ten randomly chosen elements. We see that for this example the plotted curves are within a vertical

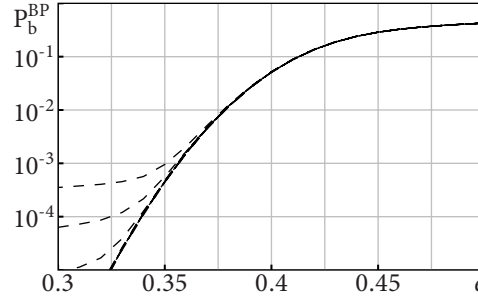


Figure 3.32: The bit erasure probability of ten random samples from $\text{LDPC}(512x^3, 256x^6) \triangleq \text{LDPC}(512, x^2, x^5)$.

distance of roughly $\delta \approx 10^{-3}$: all samples follow the “main” curve up to some point. At this point, which depends on the sample, the curve of the individual sample flattens out. We will see in Section 3.23 that the main curve is due to large-sized decoding failures (i.e., errors whose support size is a linear fraction of the blocklength) and we will give an analytic characterization of this curve. On the other hand, as we will discuss in Section 3.24, the error floor is due to certain “weaknesses” in the graph which typically can be *expurgated*. We derive the exact limiting distribution of the error floor in Lemma 3.167. \diamond

We do not prove Theorem 3.30 here. Rather this is done in a much broader context in Appendix C, where a variety of probabilistic tools and theorems are discussed which are useful in the context of message-passing coding. The main idea behind Theorem 3.30 is easy to explain: a message-passing decoder is a *local* algorithm. This means that local changes in the graph only affect local decisions. Consider a code G and some small modification, e.g., switch the endpoints of two randomly chosen edges. Because of the local nature of the message-passing algorithm this switch has (in the limit of large blocklengths) only a negligible effect on the resulting performance. Since, finally, LDPC codes have only a linear number of edges, any two codes can be converted into each other by a linear number of such elementary steps, each of which has only a small effect on its performance.

In the above theorem we have not given any explicit constants. Such constants can be furnished, and indeed they are given in Appendix C. Unfortunately though, even the best constants which have been proved to date can not explain the actual empirically observed tight concentration. Theorem 3.30 should therefore be thought more as a moral support for the approach taken, rather than a relevant engineering tool by which to judge the performance.

§3.7. COMPUTATION GRAPH AND TREE ENSEMBLE

In the previous section we have reduced the analysis already in two essential ways. First, we can assume that the all-zero word was transmitted, and second, we only need to find the ensemble-average performance. Assuming these simplifications, how can we determine the performance of LDPC codes under BP decoding?

§3.7.1. COMPUTATION GRAPH

Message passing takes place on the local neighborhood of a node/edge. As a first step we characterize this neighborhood.

EXAMPLE 3.33 (COMPUTATION GRAPH – NODE AND EDGE PERSPECTIVE). Consider again the parity-check matrix H given in (3.9). Its Tanner graph is shown in Figure 3.10. Let us focus on the decoding process for bit x_1 assuming that two rounds of message passing are performed. By convention, since no real processing is done in iteration zero, we do not count iteration zero. Recall from the description of the decoder that the decision for bit x_1 is based on the messages received from its adjoined check nodes c_5 , c_8 , and c_9 . These in turn process the information received from their *other* neighbors to form their opinion. E.g., the outgoing message of c_5 is a function of the messages arriving from x_2 , x_5 , x_{13} , x_{17} , and x_{19} . If we unroll this dependency structure for bit x_1 we arrive at the *computation graph* depicted in Figure 3.34. The figure depicts the computation graph for two iterations. With some abuse of notation we say that the computation graph has *height 2*. It is rooted in the variable node x_1 , it is bipartite, i.e., each edge connects a variable node to a check node, and all leafs are variable nodes. This computation graph is depicted as a tree, but in fact *it is not*: several of the variable and check nodes appear *repeatedly*. E.g., x_3 appears as a child of both c_8 and c_9 . Therefore, more properly, this computation graph should be drawn as a rooted graph in which each distinct node appears only once.

The above graph is a computation graph from a *node perspective* since we start from a variable node. We can also start from an *edge* and unravel the dependencies of the message sent along this edge. We call the result the computation graph from an *edge perspective*. In Figure 3.34 the resulting computation graph of height 2 for

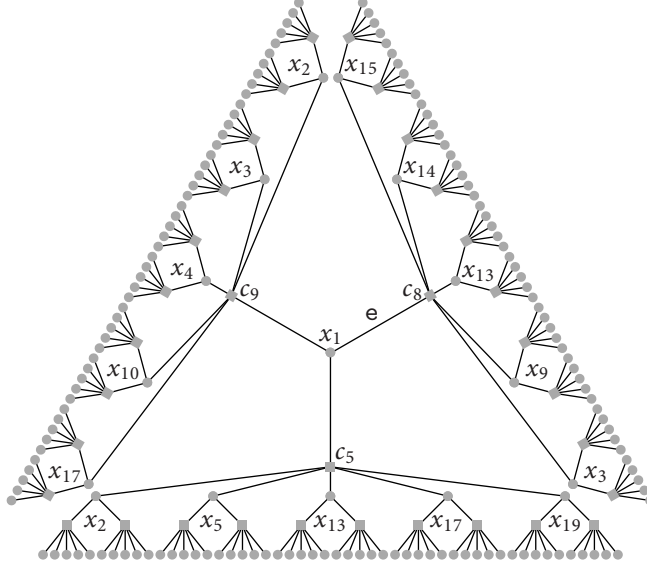


Figure 3.34: The computation graph of height 2 (two iterations) for bit x_1 and the code $C(H)$ for H given in (3.9). The computation graph of height 2 (two iterations) for edge e is the subtree consisting of edge e , variable node x_1 and the two subtrees rooted in check nodes c_5 and c_9 .

edge e is shown as well. It is the subtree consisting of variable node x_1 and the two subtrees rooted in check nodes c_5 and c_9 . \diamond

DEFINITION 3.35 (COMPUTATION GRAPH ENSEMBLE – NODE AND EDGE PERSPECTIVE). Consider the ensemble LDPC (n, λ, ρ) . The associated ensemble of computation graphs of height ℓ from a node perspective, denote it by $\hat{\mathcal{C}}_\ell(n, \lambda, \rho)$, is defined as follows. To sample from this ensemble, pick a graph G from LDPC (n, λ, ρ) uniformly at random and draw the computation graph of height ℓ of a randomly chosen variable node of G . Each such computation graph, call it T , is an *unlabeled rooted graph* in which each distinct node is drawn exactly once. The ensemble $\hat{\mathcal{C}}_\ell(n, \lambda, \rho)$ consist of the set of such computation graphs together with the probabilities $P\{T \in \hat{\mathcal{C}}_\ell(n, \lambda, \rho)\}$ which are induced by the above sampling procedure.

In the same way, to sample from the ensemble of computation graphs from an edge perspective, denote it by $\tilde{\mathcal{C}}_\ell(n, \lambda, \rho)$, pick randomly an edge e and draw the computation graph of e of height ℓ in G . Since $\hat{\mathcal{C}}_\ell(n, \lambda, \rho)$ and $\tilde{\mathcal{C}}_\ell(n, \lambda, \rho)$ share many properties it is convenient to be able to refer to both of them together. In this case we write $\mathcal{C}_\ell(n, \lambda, \rho)$. ∇

EXAMPLE 3.36 ($\hat{\mathcal{C}}_1(n, \lambda(x) = x, \rho(x) = x^2)$). In this simple example every vari-

able node has two outgoing edges and every check node has three attached edges. Figure 3.37 shows the six elements of this ensemble together with their associated probabilities $P\{T \in \mathring{\mathcal{C}}_1(n, x, x^2)\}$ as a function of n . All these probabilities behave like $O(1/n)$, except for the tree in the top row which asymptotically has probability one. Also shown is the conditional probability of error $P_b^{\text{BP}}(T, \epsilon)$. This is the probability of error which we incur if we decode the root node of the graph T assuming that transmission takes place over the $\text{BEC}(\epsilon)$ and assuming that we perform BP decoding for one iteration. \diamond





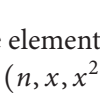
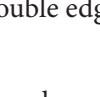
T	$P\{T \in \mathring{\mathcal{C}}_1(n, x, x^2)\}$	$P_b^{\text{BP}}(T, \epsilon)$
	$\frac{(2n-6)(2n-8)}{(2n-1)(2n-5)}$	$\epsilon(1 - (1 - \epsilon)^2)^2$
	$\frac{2(2n-6)}{(2n-1)(2n-5)}$	$\epsilon^2(1 - (1 - \epsilon)^2)$
	$\frac{1}{(2n-1)(2n-5)}$	ϵ^3
	$\frac{4(2n-6)}{(2n-1)(2n-5)}$	$\epsilon^2 + \epsilon^3(1 - \epsilon)$
	$\frac{2}{(2n-1)(2n-5)}$	$\epsilon(1 - (1 - \epsilon)^2)$
	$\frac{2}{2n-1}$	ϵ^2

Figure 3.37: The elements of $\mathring{\mathcal{C}}_1(n, \lambda(x) = x, \rho(x) = x^2)$ together with their probabilities $P\{T \in \mathring{\mathcal{C}}_1(n, x, x^2)\}$ and the conditional probability of error $P_b^{\text{BP}}(T, \epsilon)$. Thick lines indicate double edges.

The operational meaning of the ensembles $\mathring{\mathcal{C}}_\ell(n, \lambda, \rho)$ and $\vec{\mathcal{C}}_\ell(n, \lambda, \rho)$ is clear: $\mathring{\mathcal{C}}_\ell(n, \lambda, \rho)$ represents the ensemble of computation graphs which the BP decoder encounters when making a decision on a randomly chosen bit from a random sample of LDPC (n, λ, ρ) , assuming the decoder performs ℓ iterations and $\vec{\mathcal{C}}_\ell(n, \lambda, \rho)$ represents the ensemble of computation graphs which the BP decoder encounters when determining the variable-to-check message sent out along a randomly chosen edge in the ℓ -th iteration.

For $T \in \mathring{\mathcal{C}}_\ell(n, \lambda, \rho)$, let $P_b^{\text{BP}}(T, \epsilon)$ denote the conditional probability of error incurred by the BP decoder, assuming that the computation graph is T . With this notation we have

$$(3.38) \quad \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)}[P_b^{\text{BP}}(G, \epsilon, \ell)] = \sum_T P\{T \in \mathring{\mathcal{C}}_\ell(n, \lambda, \rho)\} P_b^{\text{BP}}(T, \epsilon).$$

In principle the right hand side of (3.38) can be computed exactly as shown in Figure 3.37. For a fixed ℓ , there are only a finite number of elements T in $\mathcal{C}_\ell(n, \lambda, \rho)$. The probability $P\{T \in \mathcal{C}_\ell(n, \lambda, \rho)\}$ is a combinatorial quantity, independent of the channel, and it can be determined by counting. To determine $P_b^{\text{BP}}(T, \epsilon)$ proceed as follows: recall that we can assume that the all-zero word was transmitted. Therefore, assume that all variable nodes of T are initially labeled with zero. Each such label is now erased with probability ϵ , where the choice is independent for each node. For each resulting constellation of erasures the root node can either be determined (by the BP decoder) or not. We get $P_b^{\text{BP}}(T, \epsilon)$ if we sum over all possible erasure constellations with their respective probabilities. If we perform this calculation for the example shown in Figure 3.37, the exact expression is not too revealing but if we expand the result in powers of $1/n$ we get

$$\begin{aligned} \mathbb{E}_{\text{LDPC}(n,x,x^2)}[P_b^{\text{BP}}(G, \epsilon, \ell = 1)] &= \epsilon(1 - (1 - \epsilon)^2)^2 + \\ (3.39) \quad &\epsilon^2(3 - 12\epsilon + 13\epsilon^2 - 4\epsilon^3)/n + O(1/n^2). \end{aligned}$$

We see that for increasing blocklengths the expected bit erasure probability after one iteration converges to the constant value $\epsilon(1 - (1 - \epsilon)^2)^2$. This is equal to the conditional erasure probability $P_b^{\text{BP}}(T, \epsilon)$ of the tree-like computation graph shown in the top of Figure 3.37. This result is not surprising: from Figure 3.37 we see that this computation graph has essentially (up to factors of order $O(1/n)$) probability one and that the convergence speed to this asymptotic value is therefore of order $1/n$.

Although this approach poses no conceptual problems, it quickly becomes computationally infeasible since the number of computation graphs grows exponentially in the number of iterations. Faced with these difficulties, we start with a simpler task – the determination of the limiting (in the blocklength) performance – we will come back to the finite-length analysis in Section 3.22. We will see that in this limit the repetitions in the computation graphs vanish and that the limiting performance can be characterized in terms of a recursion. In order to give a clean setting for this recursion, and also to introduce concepts which are important for the general case, we start by giving a formal definition of the limiting objects.

§3.7.2. TREE ENSEMBLE

For a fixed number of iterations and increasing blocklengths, it is intuitive that fewer and fewer cycles occur in the corresponding computation graphs. In fact, in the limit of infinitely long blocklengths the computation graph becomes a tree with probability one and each subtree of a computation graph tends to an independent sample whose distribution is determined only by the degree distribution pair (λ, ρ) .

DEFINITION 3.40 (TREE ENSEMBLES – NODE AND EDGE PERSPECTIVE). The tree ensembles $\vec{\mathcal{T}}_\ell(\lambda, \rho)$ and $\vec{\mathcal{T}}_\ell(\lambda, \rho)$ are the *asymptotic versions* of the computation graph ensembles $\vec{\mathcal{C}}_\ell(\lambda, \rho)$ and $\vec{\mathcal{C}}_\ell(\lambda, \rho)$. We start by describing $\vec{\mathcal{T}}_\ell(\lambda, \rho)$. Each element of $\vec{\mathcal{T}}_\ell(\lambda, \rho)$ is a bipartite tree rooted in a variable node. The ensemble $\vec{\mathcal{T}}_0(\lambda, \rho)$ contains a single element – the trivial tree consisting only of the root variable node, and it will serve as our anchor. Let $L(i)$ denote a bipartite tree rooted in a variable node which has i (check node) children and, in the same manner, let $R(i)$ denote a bipartite tree rooted in a check node which has i (variable node) children as shown in Figure 3.41. To sample from $\vec{\mathcal{T}}_\ell(\lambda, \rho)$, $\ell \geq 1$, first sample an element from $\vec{\mathcal{T}}_{\ell-1}(\lambda, \rho)$. Next substitute each of its leaf variable nodes with a random element from $\{L(i)\}_{i \geq 1}$, where $L(i)$ is chosen with probability λ_{i+1} . Finally, substitute each of its leaf check nodes with a random element from $\{R(i)\}_{i \geq 1}$, where $R(i)$ is chosen with probability ρ_{i+1} . The above definition implies the following recursive decomposition. In order to sample from $\vec{\mathcal{T}}_\ell(\lambda, \rho)$, sample from $\vec{\mathcal{T}}_i(\lambda, \rho)$, $0 \leq i \leq \ell$, and replace each of its (variable) leaf nodes by independent samples from $\vec{\mathcal{T}}_{\ell-i}(\lambda, \rho)$. This recursive structure is the key to the analysis of BP decoding. The description



Figure 3.41: Examples of basic trees, left $L(5)$ and right $R(7)$.

of $\vec{\mathcal{T}}_\ell(\lambda, \rho)$ differs from the one of $\vec{\mathcal{T}}_\ell(\lambda, \rho)$ only in the probabilistic choice of the root node. Again, $\vec{\mathcal{T}}_0(\lambda, \rho)$ contains a single element – the trivial tree consisting only of the root variable node. To sample from $\vec{\mathcal{T}}_1(\lambda, \rho)$, first choose an element of $\{L(i)\}_{i \geq 1}$, where element $L(i)$ has probability L_i (this is the difference to the previous definition). Next, substitute each of its leaf check nodes with a random element from $\{R(i)\}_{i \geq 1}$, where $R(i)$ is chosen with probability ρ_{i+1} . For all further steps we proceed as in the case of the ensemble $\vec{\mathcal{T}}$. It follows that we have again the following recursive decomposition. In order to sample from $\vec{\mathcal{T}}_\ell(\lambda, \rho)$, sample from $\vec{\mathcal{T}}_i(\lambda, \rho)$, $1 \leq i \leq \ell$, and replace each of its (variable) leaf nodes by independent samples from $\vec{\mathcal{T}}_{\ell-i}(\lambda, \rho)$. As for the computation graph ensembles, we apply the same convention and write $\mathcal{T}_\ell(\lambda, \rho)$ if the statement refers equally to the node or the edge tree ensemble. ∇

EXAMPLE 3.42 ($\mathcal{T}_\ell(\lambda(x) = x^{1-1}, \rho(x) = x^{r-1})$ ENSEMBLE). The tree ensembles is particularly simple for the regular case. Then each $\vec{\mathcal{T}}_\ell(\lambda(x) = x^{1-1}, \rho(x) = x^{r-1})$ consists of a single element, a bipartite graph of height ℓ , rooted in a variable node, where each variable node has $1 - 1$ check-node children and each check-node has

$r - 1$ variable node children. The same is true for $\hat{\mathcal{T}}_\ell(\lambda(x) = x^{1-1}, \rho(x) = x^{r-1})$, except that the root variable node has 1 check-node children. \diamond

EXAMPLE 3.43 ($\hat{\mathcal{T}}_\ell(\lambda(x) = \frac{1}{2}x + \frac{1}{2}x^2, \rho(x) = \frac{1}{5}x^3 + \frac{4}{5}x^4)$ ENSEMBLE). As discussed before, $\hat{\mathcal{T}}_0(\lambda, \rho)$ consists of a single element – a root variable node. The twelve elements of $\hat{\mathcal{T}}_1(\lambda, \rho)$ are shown in Figure 3.44, together with their probabilities. (Note that $\lambda(x) = 1/2x + 1/2x^2$ implies that $L(x) = 3/5x^2 + 2/5x^3$.) \diamond

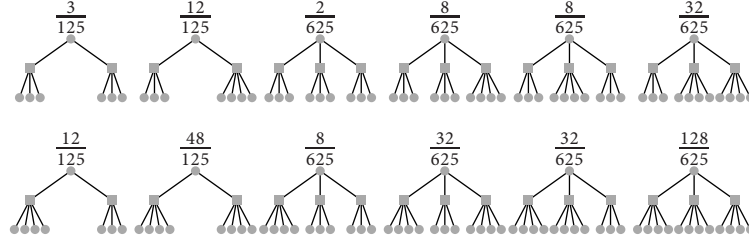


Figure 3.44: The twelve elements of $\hat{\mathcal{T}}_1(\lambda, \rho)$ together with their probabilities.

As in the case of the computation graph ensembles, we can associate with each element of $\hat{\mathcal{T}}_\ell(\lambda, \rho)$ a conditional probability of error: we imagine that each variable node is initially labeled with zero, that each label is then erased with probability ϵ by the channel, where erasures are independent, and that the BP decoder tries to determine the root node. In terms of these conditional probabilities of error we can write the probability of error of the tree ensemble as

$$P_{\hat{\mathcal{T}}_\ell(\lambda, \rho)}^{\text{BP}}(\epsilon) = \sum_{\mathbf{T}} P\{\mathbf{T} \in \hat{\mathcal{T}}_\ell(\lambda, \rho)\} P_{\mathbf{T}}^{\text{BP}}(\epsilon).$$

DEFINITION 3.45 (TREE CODE). Let $C(\mathbf{T}_\ell)$ denote the set of valid codewords on \mathbf{T}_ℓ , i.e., the set 0/1 assignments on the variables contained in \mathbf{T}_ℓ which fulfill the constraints on the tree. Further, let $C^{0/1}(\mathbf{T}_\ell)$ denote the valid codewords on \mathbf{T}_ℓ such that the root variable node is 0/1. Clearly, $C^0(\mathbf{T}_\ell)$ and $C^1(\mathbf{T}_\ell)$ are disjoint and their union equals $C(\mathbf{T}_\ell)$. Finally, we need the subset of $C^1(\mathbf{T}_\ell)$ consisting only of the *minimal codewords*, denote this set by $C_{\min}^1(\mathbf{T}_\ell)$: a codeword in $C_{\min}^1(\mathbf{T}_\ell)$ has a 1 at the root node and then for each of its connected check nodes *exactly one* of its child variable nodes is also 1. This continues until we have reached the boundary. Figure 3.46 shows a tree with a non-minimal (left) and a minimal (right) assignment. ∇

Discussion: Consider a code C and the computation graph \mathbf{T} (for an edge or a node) for a fixed number of iterations. This computation graph may or may not be a tree. Project the global codewords of C onto the set of variables contained in \mathbf{T} .

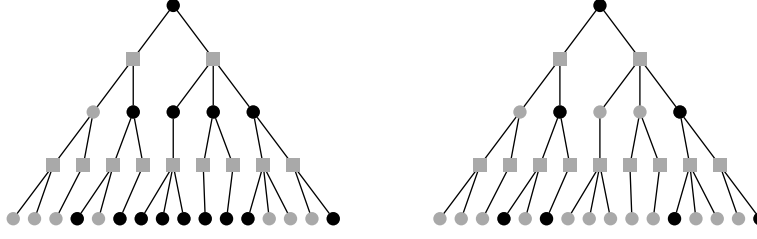


Figure 3.46: Left: An element of $C^1(T_2)$ for a given tree T_2 . Right: A minimal such element, i.e., an element of $C_{\min}^1(T_2)$. Every check node has either no connected variables of value 1 or exactly two such neighbors. Black and gray circles indicate variables with associated value 1, respectively, 0.

It can happen that this set of projections is a strict subset of $C(T)$. This happens if and only if any of the positions indexed by T are not proper and so are permanently fixed to 0 by global constraints which do not appear in the set of local constraints that define T . The BP decoder is oblivious to such global constraints and operates only on the local constraints. A MAP decoder which sees only the locally received word but is aware of the global constraints can therefore sometimes do better than the BP decoder working on the same computation graph even if this computation graph is a tree. But, as the next lemma shows, for large blocklengths, the difference is negligible. Also, a MAP decoder which is only aware of the local structure has an identical performance to a BP decoder assuming that T is a tree.

LEMMA 3.47 (REGULAR ENSEMBLES ARE PROPER). Consider the $(1, r)$ -regular ensembles of length n . Let G be chosen uniformly at random from this ensemble. Then the probability that the i -th position of G is not proper tends to zero as n tends to infinity.

§3.8. TREE CHANNEL AND CONVERGENCE TO TREE CHANNEL

§3.8.1. TREE CHANNEL

DEFINITION 3.48 ($(\mathcal{T}_\ell, \epsilon)$ -TREE CHANNEL). Given the BEC channel characterized by its erasure probability ϵ and a tree ensemble $\mathcal{T}_\ell = \mathcal{T}_\ell(\lambda, \rho)$, we define the associated $(\mathcal{T}_\ell, \epsilon)$ -tree channel. The channel takes binary input $X \in \{0, 1\}$ with uniform probability. The output of the channel is constructed as follows. Given X , first pick T from \mathcal{T}_ℓ uniformly at random. Next, pick a codeword from $C^0(T)$ uniformly at random if $X = 0$ and otherwise pick a codeword from $C^1(T)$ uniformly at random. As a shorthand, let us say that we pick a codeword from $C^X(T)$ uniformly at random. Transmit this codeword over the BEC(ϵ). Call the output Y . The receiver sees (T, Y)

and estimates X . Let $P_{\mathcal{T}_\ell}^{\text{BP}}(\epsilon)$ denote the resulting bit error probability, assuming that (T, Y) is processed by a BP decoder. ∇

Discussion: We know already that the error probability depends on the tree but not on the codeword that is sent. The distribution of the codeword is therefore irrelevant for the subsequent discussion. For sake of definiteness we have chosen this distribution to be the uniform one.

§3.8.2. CONVERGENCE TO TREE CHANNEL

THEOREM 3.49 (CONVERGENCE TO TREE CHANNEL). For a given degree distribution pair (λ, ρ) consider the sequence of associated ensembles LDPC (n, λ, ρ) for increasing blocklength n under ℓ rounds of BP decoding. Then

$$\lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)}[P_b^{\text{BP}}(G, \epsilon, \ell)] = P_{\mathcal{T}_\ell(\lambda, \rho)}^{\text{BP}}(\epsilon).$$

Proof. From characterization (3.38) we have

$$\lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)}[P_b^{\text{BP}}(G, \epsilon, \ell)] = \lim_{n \rightarrow \infty} \sum_{\mathbf{T}} P\{\mathbf{T} \in \mathcal{C}_\ell(n, \lambda, \rho)\} P_b^{\text{BP}}(\mathbf{T}, \epsilon).$$

Consider

$$\sum_{\mathbf{T}} P\{\mathbf{T} \in \mathcal{C}_\ell(n, \lambda, \rho)\} P_b^{\text{BP}}(\mathbf{T}, \epsilon) - \sum_{\mathbf{T}} P\{\mathbf{T} \in \mathcal{T}_\ell(\lambda, \rho)\} P_b^{\text{BP}}(\mathbf{T}, \epsilon).$$

Since the conditional probability of error is identical, the above difference can be written as

$$\sum_{\mathbf{T}} (P\{\mathbf{T} \in \mathcal{C}_\ell(n, \lambda, \rho)\} - P\{\mathbf{T} \in \mathcal{T}_\ell(\lambda, \rho)\}) P_b^{\text{BP}}(\mathbf{T}, \epsilon).$$

The proof now follows by observing that for each \mathbf{T} ,

$$\lim_{n \rightarrow \infty} P\{\mathbf{T} \in \mathcal{C}_\ell(n, \lambda, \rho)\} = P\{\mathbf{T} \in \mathcal{T}_\ell(\lambda, \rho)\}.$$

E.g., we see in Figure 3.37 that in the limit of infinite blocklengths the only computation graph with a positive probability is the tree shown in the top row. In the general case, any computation graph which contains at least one repetition (edge or node) has a probability that tends to zero at a speed of at least $1/n$. Finally, this convergence is uniform since there are only a finite number of possible graphs of a fixed height. \square

§3.9. DENSITY EVOLUTION

Theorem 3.49 asserts that in the limit of large blocklengths the average performance of an ensemble LDPC (n, λ, ρ) converges to the performance of the corresponding tree channel.

THEOREM 3.50 (PERFORMANCE OF TREE CHANNEL). Consider a degree distribution pair (λ, ρ) with associated normalized variable degree distribution from a node perspective $L(x)$. Let ϵ be the channel parameter, $\epsilon \in [0, 1]$. Define $x_{-1} = 1$ and for $\ell \geq 0$ let

$$(3.51) \quad x_\ell \triangleq \epsilon \lambda (1 - \rho(1 - x_{\ell-1})).$$

Then for $\ell \geq 0$

$$P_{\vec{T}_\ell}^{\text{BP}}(\epsilon) = x_\ell, \quad P_{\vec{T}_\ell}^{\text{BP}}(\epsilon) = \epsilon L(1 - \rho(1 - x_{\ell-1})).$$

Proof. Consider first $P_{\vec{T}_\ell}^{\text{BP}}(\epsilon)$. By definition, the initial variable-to-check message is equal to the received message which is an erasure message with probability ϵ . It follows that $P_{\vec{T}_0}^{\text{BP}}(\epsilon) = \epsilon$, as claimed. We use induction. Assume that $P_{\vec{T}_\ell}^{\text{BP}}(\epsilon) = x_\ell$. Consider $P_{\vec{T}_{\ell+1}}^{\text{BP}}(\epsilon)$. We start with the check-to-variable messages in the $(\ell + 1)$ -th iteration. Recall that by definition of the algorithm a check-to-variable message emitted by a check node of degree i along a particular edge is the erasure message if any of the $(i - 1)$ incoming messages is an erasure. By assumption, each such message is an erasure with probability x_ℓ and all messages are independent, so that the probability that the outgoing message is an erasure is equal to $1 - (1 - x_\ell)^{i-1}$. Since the edge has probability ρ_i to be connected to a check node of degree i it follows that the expected erasure probability of a check-to-variable message in the $(\ell + 1)$ -th iteration is equal to $\sum_i \rho_i (1 - (1 - x_\ell)^{i-1}) = 1 - \rho(1 - x_\ell)$. Now consider the erasure probability of the variable-to-check messages in the $(\ell + 1)$ -th iteration. Consider an edge e which is connected to a variable node of degree i . The outgoing variable-to-check message along this edge in the $(\ell + 1)$ -th iteration is an erasure if the received value of the associated variable node is an erasure and all $(i - 1)$ incoming messages are erasure. This happens with probability $\epsilon(1 - \rho(1 - x_\ell))^{i-1}$. Averaging again over the edge degree distribution λ , we get that $P_{\vec{T}_{\ell+1}}^{\text{BP}}(\epsilon) = \epsilon \lambda(1 - \rho(1 - x_\ell)) = x_{\ell+1}$, as claimed.

The proof for $P_{\vec{T}_\ell}^{\text{BP}}(\epsilon)$ is very similar. Recall that \vec{T}_ℓ and \vec{T}_ℓ are identical except for the choice of the root node which is chosen according to the normalized node degree distribution $L(x)$ instead of the edge degree distribution $\lambda(x)$. \square

EXAMPLE 3.52 (DENSITY EVOLUTION FOR $(\lambda(x) = x^2, \rho(x) = x^5)$). For the degree distribution pair $(\lambda(x) = x^2, \rho(x) = x^5)$ we have $x_0 = \epsilon$ and for $\ell \geq 1$, $x_\ell = \epsilon(1 -$

$(1 - x_{\ell-1})^5)^2$. For, e.g., $\epsilon = 0.4$ the sequence of values of x_ℓ is 0.4, 0.34, 0.306, 0.2818, 0.2617, 0.2438, ... \diamond

Theorem 3.50 gives a precise characterization of the asymptotic performance in terms of the recursions stated in (3.51). Those recursions are termed *density evolution* equations since they describe how the erasure probability evolves as a function of the iteration number.²

§3.10. MONOTONICITY

Monotonicity either with respect to the channel parameter or with respect to the number of iterations ℓ plays a fundamental role in the analysis of density evolution. The first lemma is a direct consequence of the non-negativity of the coefficients of the polynomials $\lambda(x)$ and $\rho(x)$ and the fact that $\rho(1) = 1$. We skip the proof.

LEMMA 3.53 (MONOTONICITY OF $f(\cdot, \cdot)$). For a given degree distribution pair (λ, ρ) define $f(\epsilon, x) \triangleq \epsilon\lambda(1 - \rho(1 - x))$. Then $f(\epsilon, x)$ is increasing in both its arguments for $x, \epsilon \in [0, 1]$.

LEMMA 3.54 (MONOTONICITY WITH RESPECT TO CHANNEL). Let (λ, ρ) be a degree distribution pair and $\epsilon \in [0, 1]$. If $P_{\tilde{T}_\ell}^{\text{BP}}(\epsilon) \xrightarrow{\ell \rightarrow \infty} 0$ then $P_{\tilde{T}_\ell}^{\text{BP}}(\epsilon') \xrightarrow{\ell \rightarrow \infty} 0$ for all $\epsilon' \leq \epsilon$.

Proof. We prove the claim for $P_{\tilde{T}_\ell}^{\text{BP}}(\epsilon)$. The corresponding claim for $P_{\tilde{T}_\ell}^{\text{BP}}(\epsilon)$ can be treated in a nearly identical manner and we skip the details. Recall from Theorem 3.50 that $P_{\tilde{T}_\ell}^{\text{BP}}(\epsilon) = x_\ell(\epsilon)$, where $x_0(\epsilon) = \epsilon$, $x_\ell(\epsilon) = f(\epsilon, x_{\ell-1}(\epsilon))$, and $f(\epsilon, x) \triangleq \epsilon\lambda(1 - \rho(1 - x))$. Assume that for some $\ell \geq 0$, $x_\ell(\epsilon') \leq x_\ell(\epsilon)$. Then

$$x_{\ell+1}(\epsilon') = f(\epsilon', x_\ell(\epsilon')) \stackrel{\text{Lemma 3.53}}{\leq} f(\epsilon', x_\ell(\epsilon)) \stackrel{\text{Lemma 3.53}}{\leq} f(\epsilon, x_\ell(\epsilon)) = x_{\ell+1}(\epsilon).$$

But if $\epsilon' \leq \epsilon$, then $x_0(\epsilon') = \epsilon' \leq \epsilon = x_0(\epsilon)$ and we conclude by induction that $x_\ell(\epsilon') \leq x_\ell(\epsilon)$. So if $x_\ell(\epsilon) \xrightarrow{\ell \rightarrow \infty} 0$, then $x_\ell(\epsilon') \xrightarrow{\ell \rightarrow \infty} 0$. \square

LEMMA 3.55 (MONOTONICITY WITH RESPECT TO ITERATION). Let $\epsilon \geq 0$ and $x_0 \in [0, 1]$. For $\ell = 1, 2, \dots$ define

$$x_\ell(x_0) = f(\epsilon, x_{\ell-1}(x_0)).$$

Then $x_\ell(x_0)$ is a monotone sequence converging to the *nearest* (in the direction of monotonicity) solution of the equation $x = f(\epsilon, x)$.

²For the BEC this “density” simplifies to a probability (of erasure) but for the general case discussed in Chapter 4, density evolution really describes an evolution of densities.

Proof. If $x_0 = 0$ or $\epsilon = 0$ then $x_\ell = 0$ for $\ell \geq 1$ and the fixed point is $x = 0$. If for some $\ell \geq 1$, $x_\ell \geq x_{\ell-1}$ then $x_{\ell+1} = f(\epsilon, x_\ell) \stackrel{\text{Lemma 3.53}}{\geq} f(\epsilon, x_{\ell-1}) = x_\ell$, and the corresponding conclusion holds if $x_\ell \leq x_{\ell-1}$. This proves the monotonicity of the sequence $\{x_\ell\}_{\ell \geq 0}$.

Since for $\epsilon \geq 0$ we have $0 \leq f(\epsilon, x) \leq \epsilon$ for all $x \in [0, 1]$, it follows that x_ℓ converges to an element of $[0, \epsilon]$, call it x_∞ . By the continuity of f we have $x_\infty = f(\epsilon, x_\infty)$. It remains to show that x_∞ is the nearest (in the sense of monotonicity) fixed point. Consider a fixed point z such that $x_\ell(x_0) \leq z$ for some $\ell \geq 0$. Then $x_{\ell+1}(x_0) = f(\epsilon, x_\ell(x_0)) \stackrel{\text{Lemma 3.53}}{\leq} f(\epsilon, z) = z$, which shows that $x_\infty \leq z$. Similarly, if $x_\ell(x_0) \geq z$ for some $\ell \geq 0$ then $x_\infty \geq z$. This shows that x_ℓ can not “jump” over any fixed point and must therefore converge to the nearest one. \square

§3.11. THRESHOLD

From the density evolution equations (3.51) we see that for every non-negative integer ℓ

$$P_{\tilde{\mathcal{T}}_\ell}^{\text{BP}}(\epsilon = 0) = 0, \quad \text{but} \quad P_{\tilde{\mathcal{T}}_\ell}^{\text{BP}}(\epsilon = 1) = 1,$$

and in particular these equalities are satisfied if $\ell \rightarrow \infty$. Further, in Problem 3.16 you will show that $P_{\tilde{\mathcal{T}}_\ell}^{\text{BP}}(\epsilon) \xrightarrow{\ell \rightarrow \infty} 0$ implies $P_{\tilde{\mathcal{T}}_\ell}^{\text{BP}}(\epsilon) \xrightarrow{\ell \rightarrow \infty} 0$, and vice versa. Combined with the above monotonicity property this shows that there is a well-defined supremum of ϵ for which $P_{\tilde{\mathcal{T}}_\ell}^{\text{BP}}(\epsilon) \xrightarrow{\ell \rightarrow \infty} 0$. This supremum is called the *threshold*.

DEFINITION 3.56 (THRESHOLD OF DEGREE DISTRIBUTION PAIR). The *threshold* associated with the degree distribution pair (λ, ρ) , call it $\epsilon^{\text{BP}}(\lambda, \rho)$, is defined as

$$\epsilon^{\text{BP}}(\lambda, \rho) \triangleq \sup\{\epsilon \in [0, 1] : P_{\tilde{\mathcal{T}}_\ell(\lambda, \rho)}^{\text{BP}}(\epsilon) \xrightarrow{\ell \rightarrow \infty} 0\}. \quad \nabla$$

EXAMPLE 3.57 (THRESHOLD OF $(\lambda(x) = x^2, \rho = x^5)$). Numerical experiments show that $\epsilon^{\text{BP}}(3, 6) \approx 0.42944$. \diamond

What is the operational meaning of $\epsilon^{\text{BP}}(\lambda, \rho)$? Using an ensemble LDPC (n, λ, ρ) of sufficient length we can transmit reliably over the channel BEC(ϵ) if $\epsilon < \epsilon^{\text{BP}}(\lambda, \rho)$ but we can not hope of doing so for channel parameters exceeding this threshold. More precisely, given $\epsilon < \epsilon^{\text{BP}}(\lambda, \rho)$ there exists an iteration number ℓ so that $P_{\tilde{\mathcal{T}}_\ell}^{\text{BP}}(\epsilon)$ is below the desired bit erasure probability. Therefore, by Theorem 3.49, ensembles LDPC (n, λ, ρ) , if decoded with ℓ rounds of BP, will show a performance approaching $P_{\tilde{\mathcal{T}}_\ell}^{\text{BP}}(\epsilon)$ as the blocklength increases. Table 3.58 lists thresholds for some regular degree distribution pairs.

l	r	$r(1, r)$	$\epsilon^{\text{Sha}}(1, r)$	$\epsilon^{\text{BP}}(1, r)$
3	6	$\frac{1}{2}$	$\frac{1}{2} = 0.5$	≈ 0.4294
4	8	$\frac{1}{2}$	$\frac{1}{2} = 0.5$	≈ 0.3834
3	5	$\frac{2}{5}$	$\frac{2}{5} = 0.6$	≈ 0.5176
4	6	$\frac{1}{3}$	$\frac{2}{3} \approx 0.667$	≈ 0.5061
3	4	$\frac{1}{4}$	$\frac{3}{4} = 0.75$	≈ 0.6474

Table 3.58: Thresholds $\epsilon^{\text{BP}}(1, r)$ under BP decoding and the corresponding Shannon thresholds $\epsilon^{\text{Sha}}(1, r)$ for some regular degree distribution pairs.

§3.12. FIXED POINT CHARACTERIZATION OF THRESHOLD

The above definition of the threshold is not very convenient for the purpose of analysis. We therefore state a second equivalent definition based on the fixed points of the density evolution equations.

THEOREM 3.59 (FIXED POINT CHARACTERIZATION OF THE THRESHOLD). For a given degree distribution pair (λ, ρ) and $\epsilon \in [0, 1]$ let $f(\epsilon, x) \triangleq \epsilon\lambda(1 - \rho(1 - x))$.

[Fixed Point Characterizations of the Threshold]

- (i) $\epsilon^{\text{BP}}(\lambda, \rho) \triangleq \sup\{\epsilon \in [0, 1] : x = f(\epsilon, x) \text{ has no solution } x \text{ in } (0, 1]\}$
- (ii) $\epsilon^{\text{BP}}(\lambda, \rho) \triangleq \inf\{\epsilon \in [0, 1] : x = f(\epsilon, x) \text{ has a solution } x \text{ in } (0, 1]\}$

Proof. Let $x(\epsilon)$ be the largest solution in $[0, 1]$ to $x = f(\epsilon, x)$. Note that for any $x \in [0, 1]$ we have $0 \leq f(\epsilon, x) \leq \epsilon$. So any solution to $x = f(\epsilon, x)$ has $x \in [0, \epsilon]$. We conclude that $x(\epsilon) \in [0, \epsilon]$.

By Lemma 3.55 we have $x_\ell(\epsilon) \xrightarrow{\ell \rightarrow \infty} x(\epsilon)$. We conclude that ϵ is below the threshold only if $x(\epsilon) = 0$ and if $x(\epsilon) > 0$ then ϵ is above the threshold. \square

DEFINITION 3.60 (CRITICAL POINT). Given a degree distribution pair (λ, ρ) which has threshold ϵ^{BP} we say that x^{BP} is a *critical point* if

$$f(\epsilon^{\text{BP}}, x^{\text{BP}}) = x^{\text{BP}} \quad \text{and} \quad \left. \frac{\partial f(\epsilon^{\text{BP}}, x)}{\partial x} \right|_{x=x^{\text{BP}}} = 1.$$

In words, x^{BP} is (one of) the point(s) at which $f(\epsilon^{\text{BP}}, x) - x$ tangentially touches the horizontal axis. ∇

The fixed point characterization gives rise to the following convenient graphical method for determining the threshold. Draw $f(\epsilon, x) - x$ as a function of x , $x \in (0, 1]$. The threshold ϵ^{BP} is the largest ϵ such that the graph of $f(\epsilon, x) - x$ is negative.

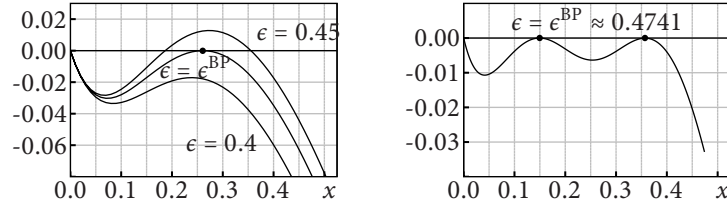


Figure 3.62: Left: Graphical determination of the threshold for $(\lambda, \rho) = (x^2, x^5)$. There is one critical point, $x^{\text{BP}} \approx 0.2606$ (black dot). Right: Graphical determination of threshold for optimized degree distribution described in Example 3.63. There are two critical points, $x_{1,2}^{\text{BP}} \approx 0.1493, 0.3571$ (two black dots).

EXAMPLE 3.61 (GRAPHICAL DETERMINATION OF THRESHOLD). The graphical determination of the threshold of $(\lambda, \rho) = (x^2, x^5)$ is shown in Figure 3.62. The graphs of $f(\epsilon, x) - x = \epsilon(1 - (1 - x)^5)^2 - x$ for the values $\epsilon = 0.4, 0.42944$, and 0.45 are depicted. We see that the supremum of all ϵ such that this plot is strictly negative for $x \in [0, 1]$ is achieved at $\epsilon^{\text{BP}} \approx 0.42944$. For this ϵ^{BP} there is one *critical* value of x , $x^{\text{BP}} \approx 0.2606$. At this point the expected decrease in the erasure fraction per iteration reaches zero so that the decoder is expected to slow down critically and come to a halt. \diamond

EXAMPLE 3.63 (OPTIMIZED ENSEMBLE). In the general case there can be more than one critical point. This happens in particular with optimized degree distributions. The more degrees we allow and the more highly optimized ensembles we consider the more simultaneous critical points we are likely to find. Consider the degree distribution pair

$$\begin{aligned}\lambda(x) &= 0.106257x + 0.486659x^2 + 0.010390x^{10} + 0.396694x^{19}, \\ \rho(x) &= 0.5x^7 + 0.5x^8.\end{aligned}$$

It has a design rate of one-half, a threshold of $\epsilon^{\text{BP}} \approx 0.4741$, and two critical points $x^{\text{BP}} \approx 0.35713565$ and $x^{\text{BP}} \approx 0.14932401$. Why do we get multiple critical points if we optimize degree distributions? In Section 3.14.4 we will see that in order to achieve capacity we need $f(\epsilon^{\text{BP}}, x) = x$ for the whole range $x \in [0, \epsilon^{\text{BP}}]$, i.e., we need equality everywhere so that all points in this range are critical points. This is called the *matching condition*. Therefore, the closer we get to capacity the more critical points we expect to see. \diamond

§3.13. STABILITY

Expanding the right hand side of (3.51) into a Taylor series around zero we get

$$(3.64) \quad x_\ell = \epsilon \lambda'(0) \rho'(1) x_{\ell-1} + O(x_{\ell-1}^2).$$

For sufficiently small x_ℓ the convergence behavior is determined by the term linear in x_ℓ . More precisely, the convergence depends on whether $\epsilon \lambda'(0) \rho'(1)$ is smaller or larger than one.

THEOREM 3.65 (STABILITY CONDITION). Assume that we are given a degree distribution pair (λ, ρ) and a real number $\epsilon, \epsilon \in [0, 1]$. Let $x_\ell(\epsilon)$ be as defined in (3.51).

[Necessity] If $\epsilon \lambda'(0) \rho'(1) > 1$ then there exists a strictly positive constant $\xi = \xi(\lambda, \rho, \epsilon)$, such that $\lim_{\ell \rightarrow \infty} x_\ell(x_0) \geq \xi$ for all $x_0 \in (0, 1)$.

[Sufficiency] If $\epsilon \lambda'(0) \rho'(1) < 1$ then there exists a strictly positive constant $\xi = \xi(\lambda, \rho, \epsilon)$, such that $\lim_{\ell \rightarrow \infty} x_\ell(x_0) = 0$ for all $x_0 \in (0, \xi)$.

Note that $\epsilon \lambda(1 - \rho(1 - 0)) = 0$ for any initial erasure fraction ϵ , so that zero is a *fixed point* of the recursion given in (3.51). Therefore, the above condition is the *stability* condition of the fixed point at zero. The most important consequence of the stability condition is the implied upper bound on the threshold

$$(3.66) \quad \epsilon^{\text{BP}}(\lambda, \rho) \leq \frac{1}{\lambda'(0) \rho'(1)}.$$

In fact, this bound also applies to MAP decoding.

LEMMA 3.67 (STABILITY CONDITION AND MAP THRESHOLD). Assume that we are given a degree distribution pair (λ, ρ) and a real number $\epsilon, \epsilon \in [0, 1]$.

[Necessity] If $\epsilon \lambda'(0) \rho'(1) > 1$ then there exists a strictly positive constant $\xi = \xi(\lambda, \rho, \epsilon)$, such that $\lim_{n \rightarrow \infty} P_b^{\text{MAP}}(n, \lambda, \rho, \epsilon) \geq \xi$.

Proof. To each element $G \in \text{LDPC}(n, \lambda, \rho)$ and each channel realization associate a “normal” graph (not bipartite), call it Γ . The nodes of Γ are the check nodes of G . The edges of Γ correspond to the degree-two variable nodes in G whose values were erased by the channel: each such variable node of degree two has exactly two outgoing edges in G and so naturally forms an edge in Γ . This connection between the bipartite graph G and the graph Γ is discussed in more detail in Section C.5.

Let $\mu = \epsilon \lambda'(0) \rho'(1)$. From Lemma C.37 and Lemma C.38 we know that if $\mu > 1$ then a positive fraction of nodes in Γ lie on cycles.

A cycle in Γ corresponds to a cycle in G so that all involved nodes are of degree two and have been erased by the channel. Such a cycle constitutes a codeword, all

of whose components have been erased. No decoder (not even a MAP decoder) can recover the value associated with these bits. Since the fraction of concerned bits is positive, we conclude that there is a positive bit erasure probability. In other words, we are transmitting above the MAP bit threshold. \square

§3.14. EXIT CHART

An EXIT chart is a helpful visualization of the asymptotic performance under BP decoding. For the BEC it is equivalent to density evolution.

§3.14.1. GRAPHICAL REPRESENTATION OF DENSITY-EVOLUTION

Consider a degree distribution pair (λ, ρ) . Recall from Section 3.12 that the asymptotic behavior of such a degree distribution pair is characterized by $f(\epsilon, x) \triangleq \epsilon\lambda(1 - \rho(1 - x))$, which represents the evolution of the fraction of erased messages emitted by variable nodes, assuming that the system is in *state* (has a current such fraction of) x and that the channel parameter is ϵ . It is helpful to represent $f(\epsilon, x)$ as the composition of two functions, one which represents the “action” of the variable nodes (which we can think of as a repetition code) and the second which describes the “action” of check nodes (a simple parity-check code). We define $v_\epsilon(x) \triangleq \epsilon\lambda(x)$ and $c(x) \triangleq 1 - \rho(1 - x)$, so that $f(\epsilon, x) = v_\epsilon(c(x))$. Recall that the condition for convergence reads $f(\epsilon, x) < x$, $\forall x \in (0, 1)$. Observe that $v_\epsilon(x)$ has an inverse for $x \geq 0$ since $\lambda(x)$ is a polynomial with non-negative coefficients. The condition for convergence can hence be written as

$$c(x) < v_\epsilon^{-1}(x), \quad x \in (0, 1).$$

This has a pleasing graphical interpretation: $c(x)$ has to lie strictly below $v_\epsilon^{-1}(x)$ over the whole range $x \in (0, 1)$. The threshold ϵ^{BP} is the supremum of all numbers ϵ for which this condition is fulfilled. The local such condition around $x = 0$, i.e., the condition for small positive values x , reads $\rho'(1) = c'(0) \leq \frac{dv_\epsilon^{-1}(x)}{dx}|_{x=0} = \frac{1}{\epsilon\lambda'(0)}$. This is the stability condition.

EXAMPLE 3.68 (GRAPHICAL REPRESENTATION FOR $(\lambda(x) = x^2, \rho(x) = x^5)$). We have $v_\epsilon^{-1}(x) = (x/\epsilon)^{1/2}$ and $c(x) = 1 - (1 - x)^5$. The curves corresponding to $v_\epsilon^{-1}(x)$ for $\epsilon = 0.35, 0.42944$, and 0.50 as well as the curve corresponding to $c(x)$ are plotted in the left picture of Figure 3.69. For $\epsilon \approx 0.42944$, $v_\epsilon^{-1}(x)$ just touches $c(x)$ for some $x \in (0, 1)$, i.e., $\epsilon^{\text{BP}} \approx 0.42944$. The picture on the right of Figure 3.69 shows the evolution of the decoding process for $\epsilon = 0.35$. \diamond

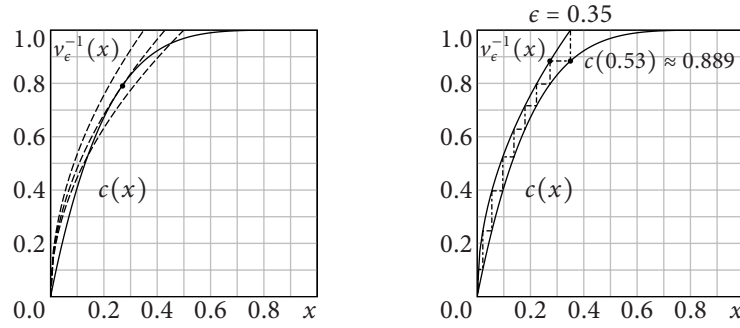


Figure 3.69: The left hand side shows the graphical determination of the threshold for $(\lambda(x) = x^2, \rho(x) = x^5)$. The function $v_\epsilon^{-1}(x) = (x/\epsilon)^{1/2}$ is shown as a dashed line for $\epsilon = 0.35$, $\epsilon = \epsilon^{\text{BP}} \approx 0.42944$, and $\epsilon = 0.5$. The function $c(x) = 1 - (1 - x)^5$ is shown as a solid line. The figure on the right hand side shows the evolution of the decoding process for $\epsilon = 0.35$. The initial fraction of erasure messages emitted by the variable nodes is $x = 0.35$. After half an iteration (at the output of the check nodes) this fraction has evolved to $c(x = 0.35) \approx 0.88397$. After one full iteration, i.e., at the output of the variable nodes we see an erasure fraction of $x = v_\epsilon(0.88397)$, i.e., x is the solution to the equation $0.883971 = v_\epsilon^{-1}(x)$. This process continues in the same fashion for each subsequent iteration, corresponding graphically to a *staircase* function which is bounded below by $c(x)$ and bounded above by $v_\epsilon^{-1}(x)$.

§3.14.2. EXIT FUNCTION

We will now see that both $c(x)$ as well as $v_\epsilon(x)$ have an interpretation in terms of entropy.

DEFINITION 3.70 (EXIT FUNCTION). Let C be a binary code. Let X be chosen with probability $p_X(x)$ from C and let Y denote the result of letting X be transmitted over a BEC(ϵ). The *extrinsic information transfer* (EXIT) function associated with the i -th bit of C , call it $h_i(\epsilon)$, is defined as

$$h_i(\epsilon) \triangleq H(X_i | Y_{\sim i}).$$

The *average* EXIT function is

$$h(\epsilon) \triangleq \frac{1}{n} \sum_{i=1}^n h_i(\epsilon). \quad \nabla$$

Discussion: The input parameter ϵ represents an entropy: if we assume that X_i is chosen uniformly at random from $\{0, 1\}$ and that Y_i is the result of sending X_i over

a BEC then $H(X_i | Y_i) = \epsilon H(X_i | Y_i = ?) = \epsilon$. The EXIT function therefore characterizes how entropy is transferred from input to output. You are warned that in the literature mutual information is often considered instead of entropy. Assuming that we impose a uniform prior on X_i , this differs from our definition only in a trivial way ($H(X_i | Y_{\sim i}) = 1 - I(X_i; Y_{\sim i})$).

EXAMPLE 3.71 (PARITY-CHECK CODE). Consider the binary parity-check code with parameters $C[n, n-1, 2]$ and a uniform prior on the set of codewords. By symmetry, $h_i(\epsilon) = h(\epsilon)$ for all $i \in [n]$ and

$$h(\epsilon) = 1 - (1 - \epsilon)^{n-1}.$$

A check node of degree i in the Tanner graph represents a parity-check code $C[i, i-1, 2]$. Let the channel parameter of the BEC be x . From above the associated EXIT function is $1 - (1 - x)^{i-1}$. The function $c(x)$ depicted in Figure 3.69 is the *average* over the set of EXIT functions corresponding to check nodes of degree i , where the average is with respect to the edge degree distribution: indeed, $\sum_i \rho_i (1 - (1 - x)^{i-1}) = 1 - \rho(1 - x) = c(x)$. \diamond

EXAMPLE 3.72 (REPETITION CODE). Consider the binary repetition code $C[n, 1, n]$ with a uniform prior on the set of codewords. By symmetry, $h_i(\epsilon) = h(\epsilon)$ for all $i \in [n]$ and

$$h(\epsilon) = \epsilon^{n-1}. \quad \diamond$$

EXAMPLE 3.73 ($[7, 4, 3]$ HAMMING CODE). The parity-check matrix H of the $[7, 4, 3]$ binary Hamming code is stated explicitly in Example 1.24. Assuming a uniform prior on the set of codewords, a tedious calculation reveals (see Problem 3.29) that

$$h_{\text{Ham}}(\epsilon) = 3\epsilon^2 + 4\epsilon^3 - 15\epsilon^4 + 12\epsilon^5 - 3\epsilon^6. \quad \diamond$$

Figure 3.74 depicts the EXIT functions of Examples 3.71, 3.72, and 3.73.

There are many alternative characterizations of EXIT functions, each with its own merit. Let us list the most useful ones.

LEMMA 3.75 (VARIOUS CHARACTERIZATIONS OF EXIT FUNCTIONS). Let C be a binary code. Let X be chosen with probability $p_X(x)$ from C and let Y denote the result of letting X pass over a BEC(ϵ). Let $\hat{x}_i^{\text{MAP}}(y_{\sim i})$ denote the MAP estimator function of the i -th bit given the observation $y_{\sim i}$. Then the following are equivalent:

- (i) $h_i(\epsilon) \triangleq H(X_i | Y_{\sim i})$
- (ii) $h_i(\epsilon) \triangleq \mathbb{P}\{\hat{x}_i^{\text{MAP}}(Y_{\sim i}) = ?\}$

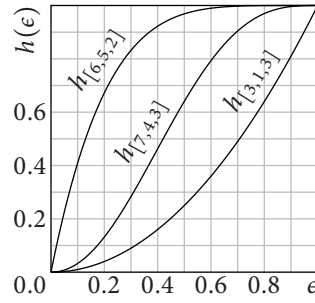


Figure 3.74: The EXIT function of the $[3, 1, 3]$ repetition code, the $[6, 5, 2]$ parity-check code, and the $[7, 4, 3]$ Hamming code.

$$(iii) \quad h(\epsilon) \triangleq \frac{dH(X|Y)}{n d\epsilon}$$

If $C[n, k]$ is a binary linear code, X is chosen with uniform probability from $C[n, k]$, and H and G is a parity, respectively, generator matrix representing C , then the two following characterizations are also equivalent:

$$(iv) \quad h_i(\epsilon) \triangleq \sum_{\mathcal{E} \subseteq [n] \setminus \{i\}} \epsilon^{|\mathcal{E}|} (1 - \epsilon)^{n-1-|\mathcal{E}|} (1 + \text{rank}(H_{\mathcal{E}}) - \text{rank}(H_{\mathcal{E} \cup \{i\}}))$$

$$(v) \quad h_i(\epsilon) \triangleq \sum_{\mathcal{E} \subseteq [n] \setminus \{i\}} \epsilon^{n-1-|\mathcal{E}|} (1 - \epsilon)^{|\mathcal{E}|} (\text{rank}(G_{\mathcal{E} \cup \{i\}}) - \text{rank}(G_{\mathcal{E}}))$$

Discussion: The second characterization states that the EXIT value equals the erasure probability of an extrinsic MAP decoder. This is useful when doing actual computations. The most fundamental characterization is the third one: it states that the (average) EXIT function equals the derivative of the conditional entropy with respect to the channel parameter divided by the blocklength n .

Proof. The proof of the equivalence of characterizations (i), (ii), (iv), and (v) is left as Problem 3.27.

In what follows we concentrate on the third characterization. Although all bits are sent through the same channel $\text{BEC}(\epsilon)$, it is convenient to imagine that bit i is sent through a BEC with parameter ϵ_i , where (by chance) $\epsilon_i = \epsilon$ for all $i \in [n]$. We claim that

$$\frac{dH(X|Y)}{d\epsilon} = \sum_{i=1}^n \frac{\partial H(X|Y)}{\partial \epsilon_i} \Big|_{\epsilon_j=\epsilon, \forall j \in [n]} = \sum_{i=1}^n \frac{\partial H(X_i|Y)}{\partial \epsilon_i} \Big|_{\epsilon_j=\epsilon, \forall j \in [n]}.$$

To see the second transition, use the chain rule to write $H(X|Y) = H(X_i|Y) + H(X_{\sim i}|X_i, Y)$. Due to the memoryless property of the channel, we further have $H(X_{\sim i}|X_i, Y) = H(X_{\sim i}|X_i, Y_{\sim i})$. But the right hand side is no longer a function

of ϵ_i . Therefore its contribution vanishes when we take the (partial) derivative with respect to ϵ_i . Finally, note that

$$H(X_i | Y) = P\{\hat{x}_i^{\text{MAP}}(Y_i) = ?\} P\{\hat{x}_i^{\text{MAP}}(Y_{\sim i}) = ?\} = \epsilon_i h_i(\epsilon),$$

so that $\frac{\partial H(X_i | Y)}{\partial \epsilon_i} = h_i(\epsilon)$ and $\frac{dH}{nd\epsilon} = h(\epsilon)$. \square

As we have seen in the proof above, it is useful to generalize to a situation where the i -th bit is sent through the channel $\text{BEC}(\epsilon_i)$. Let us assume that the individual channel parameters ϵ_i are parametrized in a differentiable way by a common parameter ϵ , i.e., $\epsilon_i = \epsilon_i(\epsilon)$. Now ϵ is no longer the channel parameter itself but if we change ϵ in a smooth manner then we can imagine that the set of individual channels $\{\text{BEC}(\epsilon_i)\}_{i=1}^n$ describes some smooth curve in “channel space”. Depending on the parametrization this description includes, e.g., the case where all channels stay fixed except for one, or the case where all channels are the same and are changed in lock-step. Characterization (iii) of Theorem 3.75 is still meaningful, i.e., we can define the (average) EXIT function as $h(\epsilon) \triangleq \frac{dH(X|Y)}{nd\epsilon}$ and the individual EXIT functions as $h_i(\epsilon) \triangleq \frac{\partial H(X_i | Y)}{\partial \epsilon_i} \frac{d\epsilon_i}{d\epsilon}$.

EXAMPLE 3.76 (EXIT FUNCTION OF VARIABLE NODES). Consider the EXIT function of a $[i + 1, 1, i]$ repetition code, where the last bit is passed through a BEC with parameter ϵ and bit 1 to bit i are passed through a BEC with parameter x . This is the case for a variable node of degree i in the Tanner graph of an LDPC code. Consider the EXIT function corresponding to one of the first i positions. We have $H(X | Y) = \epsilon \prod_{k=1}^i x_k$, where all the x_k have value equal to x . Taking the derivative with respect to e.g., x_1 and setting $x_k = x$ we get ϵx^{i-1} . If we average this over the edge degree distribution λ we get $\sum_i \lambda_i \epsilon x^{i-1} = \epsilon \lambda(x) = v_\epsilon(x)$. We conclude that $v_\epsilon(x)$ is the average of a collection of EXIT functions. \diamond

As we have seen in Examples 3.71 and 3.76, $v_\epsilon(x)$ and $c(x)$ are (averaged) EXIT functions. For this reason we call Figure 3.69 an *EXIT chart*.

§3.14.3. BASIC PROPERTIES OF EXIT FUNCTIONS

THEOREM 3.77 (DUALITY THEOREM). Let C be a binary linear code, let C^\perp be its dual and, assuming uniform priors on the codewords, let $h_i(\epsilon)$ and $h_i^\perp(\epsilon)$ denote the corresponding EXIT functions. Then

$$h_i(\epsilon) + h_i^\perp(1 - \epsilon) = 1.$$

Proof. Recall that if H is a parity-check matrix of C then it is also a generator matrix of C^\perp . The claim now follows from characterizations 4 and 5 of Lemma 3.75. \square

EXAMPLE 3.78 (REPETITION CODE AND SINGLE PARITY-CHECK CODE). The codes $[n, 1, n]$ and $[n, n-1, 2]$ are duals and from Examples 3.72 and 3.71 we know that for all $i \in [n]$ their EXIT functions are ϵ^{n-1} and $1 - (1 - \epsilon)^{n-1}$, respectively. Therefore, in agreement with Theorem 3.77,

$$\epsilon^{n-1} + 1 - (1 - (1 - \epsilon))^{n-1} = 1. \quad \diamond$$

THEOREM 3.79 (MINIMUM DISTANCE THEOREM). Let C be a binary linear code of length n with minimum distance d and with EXIT functions $h_i(\epsilon)$, $i \in [n]$. Then $h_i(\epsilon)$, $i \in [n]$, is a polynomial of minimum degree at least $d - 1$ and the average EXIT function has minimum degree exactly $d - 1$.

Proof. Consider characterization 4 of Lemma 3.75. Let \mathcal{E} be a subset of cardinality strictly less than $d - 1$. Since any $d - 1$ or less columns of H are linearly independent, it follows that $(1 + \text{rank}(H_{\mathcal{E}}) - \text{rank}(H_{\mathcal{E} \cup \{i\}}))$ is zero for any such subset \mathcal{E} . Therefore, $h_i(\epsilon)$ does not contain monomials of degree less than $d - 1$. On the other hand, if $\mathcal{E} \cup \{i\}$ is chosen to correspond to the support of a minimum distance codeword then $(1 + \text{rank}(H_{\mathcal{E}}) - \text{rank}(H_{\mathcal{E} \cup \{i\}}))$ is one and this will contribute a monomial of degree $d - 1$. Since these minimum degree terms can not be canceled by any other terms it follows that $h(\epsilon)$ has minimum degree exactly $d - 1$. \square

EXAMPLE 3.80 (REPETITION CODE, PARITY-CHECK CODE, HAMMING CODE). We see from our previous examples that the average EXIT functions of the $[n, 1, n]$ repetition code, the $[n, n-1, 2]$ single parity-check code and the $[7, 4, 3]$ Hamming code have minimum degree $n - 1$, 1 and 2 respectively, as predicted by Theorem 3.79. \diamond

The most fundamental property of EXIT functions is a direct consequence of the third characterization in Lemma 3.75. For historical reasons we label it as a theorem, even though there is really nothing to be proved.

THEOREM 3.81 (AREA THEOREM). Let C be a binary code. Let X be chosen with probability $p_X(x)$ from C and let Y denote the received word after transmission over a BEC(ϵ). To emphasize that Y depends on the channel parameter ϵ write $Y(\epsilon)$. If $h(\epsilon)$ denotes the corresponding EXIT function then

$$(3.82) \quad H(X | Y(\delta)) = n \int_0^\delta h(\epsilon) d\epsilon.$$

Proof. Using the characterization $h(\epsilon) = \frac{dH(X | Y(\epsilon))}{n d\epsilon}$ of Lemma 3.75, we have

$$\int_0^\delta \frac{dH(X | Y(\epsilon))}{d\epsilon} d\epsilon = H(X | Y(\delta)) - H(X | Y(0)) = H(X | Y(\delta)).$$

\square

Problem 3.30 discusses a combinatorial proof of this theorem which starts with the characterization of $h(\epsilon)$ given in Definition 3.70.

EXAMPLE 3.83 (AREA THEOREM APPLIED TO $[7, 4, 3]$ HAMMING CODE).

$$\int_0^1 h_{\text{Ham}}(\epsilon) d\epsilon = \int_0^1 (3\epsilon^2 + 4\epsilon^3 - 15\epsilon^4 + 12\epsilon^5 - 3\epsilon^6) d\epsilon = \frac{4}{7}. \quad \diamond$$

§3.14.4. MATCHING CONDITION

Let us go back to the EXIT chart method as depicted on the right of Figure 3.69. The area under the curve $c(x)$ equals $1 - \int \rho$ and the area to the left of the curve $v_\epsilon^{-1}(x)$ is equal to $\epsilon \int \lambda$. This is easily verified by a direct calculation, integrating the respective functions, but it also follows from (the Area) Theorem 3.81: the rate of a parity-check code of length i is $\frac{i-1}{i} = 1 - \frac{1}{i}$, so that the area under an individual EXIT function of such a code is $1 - \frac{1}{i}$. If we average over the edge degree distribution we get $1 - \sum_i \rho_i \frac{1}{i} = 1 - \int \rho$. The argument at the variable node side is more interesting. First note that the area “to the left” of the curve $v_\epsilon^{-1}(x)$ is equal to the area “under” $v_\epsilon(x)$. By definition, if we integrate the average EXIT function then we get the difference of entropies at the two endpoints of the “path.” For the EXIT function corresponding to a variable node one position is fixed to the channel $\text{BEC}(\epsilon)$ and the other ones go from “no information” to “perfect channel.” If all the other positions have “no information” then the uncertainty is ϵ , if all the other positions see a “perfect channel” then the uncertainty is 0. The difference is therefore ϵ . The EXIT function is the average of the individual EXIT functions: the EXIT function associated with the fixed position is zero (since the input does not change) and the remaining i individual EXIT functions are equal by symmetry. We conclude that the integral under one of these EXIT functions is $\frac{\epsilon}{i}$. If we average with respect to the edge degree distribution λ this gives us $\epsilon \int \lambda$.

We know that a necessary condition for successful BP decoding is that these two areas do not overlap (the two curves $v_\epsilon^{-1}(x)$ and $c(x)$ must not cross). Since the total area equals 1, we get the necessary condition (for successful BP decoding) $(\epsilon \int \lambda) + (1 - \int \rho) \leq 1$. Rearranging terms, this is equivalent to the condition

$$1 - C_{\text{Sha}} = \epsilon \leq \frac{\int \rho}{\int \lambda} = 1 - r(\lambda, \rho).$$

In words, the design rate $r(\lambda, \rho)$ of any LDPC ensemble which, for increasing block-lengths, allows successful decoding over the $\text{BEC}(\epsilon)$ can not surpass the Shannon limit $1 - \epsilon$.

The final result (namely that transmission above capacity is not possible) is trivial, but the method of proof shows how capacity enters in the calculation of the

performance under BP decoding and it shows that in order for the design rate to achieve capacity the component curves have to be perfectly matched. In the next section we construct capacity-achieving degree distributions. We use the matching condition $\epsilon\lambda(1 - \rho(1 - x)) = x$ as our starting point. This matching condition also explains why optimized ensembles tend to have many critical points as we discussed in Example 3.63.

§3.15. CAPACITY-ACHIEVING DEGREE DISTRIBUTIONS

Consider a pair of degree distributions (λ, ρ) of design rate $r = r(\lambda, \rho)$ and with threshold $\epsilon^{\text{BP}} = \epsilon^{\text{BP}}(\lambda, \rho)$. By Shannon, we must have $r \leq (1 - \epsilon^{\text{BP}})$, and it is natural to use the following definition of the (multiplicative) gap as a measure of performance.

DEFINITION 3.84 (MULTIPLICATIVE GAP). Let (λ, ρ) be a degree distribution pair with rate $r = r(\lambda, \rho)$ and threshold $\epsilon^{\text{BP}} = \epsilon^{\text{BP}}(\lambda, \rho)$. Further, let $\delta = \delta(\lambda, \rho)$ be the unique non-negative number such that $r = (1 - \delta)(1 - \epsilon^{\text{BP}})$, i.e., $\delta \triangleq \frac{1 - \epsilon^{\text{BP}} - r}{1 - \epsilon^{\text{BP}}}$. We then say that (λ, ρ) *achieves a fraction $(1 - \delta)$ of capacity*. Equivalently, we say that (λ, ρ) has a *multiplicative gap δ* . ∇

Unfortunately, as the next theorem shows, no fixed pair (λ, ρ) can have zero gap. The proof of this theorem is the topic of Problem 3.18.

THEOREM 3.85 (LOWER BOUND ON GAP). Let (λ, ρ) be a degree distribution pair of design rate r , $r \in (0, 1)$, and with average check-node degree r_{avg} . Then

$$\delta(\lambda, \rho) \geq \frac{r^{r_{\text{avg}}-1}(1 - r)}{1 + r^{r_{\text{avg}}-1}(1 - r)}.$$

The best we can therefore hope for is to construct a *sequence* of degree distribution pairs that achieve capacity.

DEFINITION 3.86. We say that a sequence $\{(\lambda^{(N)}, \rho^{(N)})\}_{N \geq 1}$ *achieves capacity* on the BEC(ϵ) if

$$(3.87) \quad \begin{aligned} \lim_{N \rightarrow \infty} r(\lambda^{(N)}, \rho^{(N)}) &= 1 - \epsilon, \text{ and} \\ \lim_{N \rightarrow \infty} \delta(\lambda^{(N)}, \rho^{(N)}) &= 0. \end{aligned}$$

Note that (3.87) implies that $\epsilon^{\text{BP}}(\lambda^{(N)}, \rho^{(N)})$ converges to ϵ . ∇

If you are mainly interested in how in *practice* we can find degree distributions that allow reliable transmission close to capacity you can fast forward to Section 3.18. In this section we want to *prove* that capacity-achieving degree distributions exist.

Our plan for constructing such sequences is the following: let \mathcal{D} be the set of non-zero functions $\hat{y} : [0, 1) \rightarrow \mathbb{R}^+$ which are analytic around zero with non-negative expansion, and let \mathcal{N} be the set of functions $\hat{y} \in \mathcal{D}$ so that $\hat{y}(1) = 1$. As a main ingredient we need two parametrized functions: for all α in some suitable chosen subset of \mathbb{R}^+ , let $\rho_\alpha(x)$ be an element of \mathcal{N} and let $\hat{\lambda}_\alpha(x)$ be an element of \mathcal{D} . Assume further a perfect matching,

$$(3.88) \quad \hat{\lambda}_\alpha(1 - \rho_\alpha(1 - x)) = x, \quad \forall x \in [0, 1).$$

Let $\hat{\lambda}_\alpha^{(N)}(x)$ denote the function consisting of the first N terms of the Taylor series expansion of $\hat{\lambda}_\alpha(x)$ (up to and including the term x^{N-1}). Clearly, $\hat{\lambda}_\alpha^{(N)}(x) \in \mathcal{D}$. For N sufficiently large, $\hat{\lambda}_\alpha^{(N)}(1) > 0$, so that we can define the *normalized* function $\lambda_\alpha^{(N)}(x) \triangleq \frac{\hat{\lambda}_\alpha^{(N)}(x)}{\hat{\lambda}_\alpha^{(N)}(1)}$, $\lambda_\alpha^{(N)}(x) \in \mathcal{N}$. The pair $(\lambda_\alpha^{(N)}(x), \rho_\alpha)$ is our candidate degree distribution pair to achieve the desired gap δ . It remains to see if we can choose N and α in a suitable way. As function of N and α , the rate can be expressed as (see (3.19))

$$r(\alpha, N) = 1 - \frac{\int_0^1 \rho_\alpha(x) dx}{\int_0^1 \lambda_\alpha^{(N)}(x) dx} = 1 - \hat{\lambda}_\alpha^{(N)}(1) \frac{\int_0^1 \rho_\alpha(x) dx}{\int_0^1 \hat{\lambda}_\alpha^{(N)}(x) dx}.$$

From the non-negativeness of the coefficients of the expansion of $\hat{\lambda}_\alpha(x)$ it follows that for $x \in [0, 1]$,

$$\hat{\lambda}_\alpha(x) \geq \hat{\lambda}_\alpha^{(N)}(x),$$

so that from (3.88)

$$x = \hat{\lambda}_\alpha(1 - \rho_\alpha(1 - x)) \geq \hat{\lambda}_\alpha^{(N)}(1) \lambda_\alpha^{(N)}(1 - \rho_\alpha(1 - x)).$$

It follows that $\epsilon^{\text{BP}}(\alpha, N) \geq \hat{\lambda}_\alpha^{(N)}(1)$. Therefore,

$$\delta(\alpha, N) = \frac{1 - \epsilon^{\text{BP}}(\alpha, N) - r}{1 - \epsilon^{\text{BP}}(\alpha, N)} \leq \frac{\hat{\lambda}_\alpha^{(N)}(1)}{1 - \hat{\lambda}_\alpha^{(N)}(1)} \left[\frac{\int_0^1 \rho_\alpha(x) dx}{\int_0^1 \hat{\lambda}_\alpha^{(N)}(x) dx} - 1 \right].$$

Our strategy is now clear. We choose (if possible) $\alpha = \alpha(N)$ in such a way that

$$1 - \hat{\lambda}_\alpha^{(N)}(1) \frac{\int_0^1 \rho_\alpha(x) dx}{\int_0^1 \hat{\lambda}_\alpha^{(N)}(x) dx} \xrightarrow{N \rightarrow \infty} 1 - \epsilon, \quad \frac{\hat{\lambda}_\alpha^{(N)}(1)}{1 - \hat{\lambda}_\alpha^{(N)}(1)} \left[\frac{\int_0^1 \rho_\alpha(x) dx}{\int_0^1 \hat{\lambda}_\alpha^{(N)}(x) dx} - 1 \right] \xrightarrow{N \rightarrow \infty} 0.$$

If this can be done we get a sequence of capacity achieving degree distribution pairs.

EXAMPLE 3.89 (CHECK-CONCENTRATED DEGREE DISTRIBUTION). For $\alpha \in (0, 1)$ so that $1/\alpha \in \mathbb{N}$, choose

$$\hat{\lambda}_\alpha(x) \triangleq 1 - (1-x)^\alpha = \sum_{i=1}^{\infty} \binom{\alpha}{i} (-1)^{i-1} x^i, \text{ and}$$

$$\rho_\alpha(x) \triangleq x^{\frac{1}{\alpha}}.$$

Recall that for $\alpha \in \mathbb{R}$ and $i \in \mathbb{N}$, $\binom{\alpha}{i} (-1)^{i-1}$ is defined as

$$\binom{\alpha}{i} (-1)^{i-1} \triangleq \frac{\alpha(\alpha-1)\dots(\alpha-i+1)}{i!} (-1)^{i-1} = \frac{\alpha}{i} \left(1 - \frac{\alpha}{i-1}\right) \dots (1-\alpha).$$

Since $\alpha \in (0, 1)$, all coefficients of the expansion of $\hat{\lambda}_\alpha(x)$ are non-negative. We have

$$\int_0^1 \hat{\lambda}_\alpha^{(N)}(x) dx = \sum_{i=1}^{N-1} \binom{\alpha}{i} \frac{(-1)^{i-1}}{i+1} = \frac{\alpha - \binom{\alpha}{N} (-1)^{N-1}}{\alpha + 1},$$

$$\hat{\lambda}_\alpha^{(N)}(1) = \sum_{i=1}^{N-1} \binom{\alpha}{i} (-1)^{i-1} = 1 - \frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1},$$

and

$$\int_0^1 \rho_\alpha(x) dx = \frac{\alpha}{1+\alpha}.$$

It follows that

$$r(\alpha, N) = \frac{\frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1} (1 - 1/N)}{1 - \frac{1}{N} \frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1}}, \quad \delta(\alpha, N) \leq \frac{1 - \frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1}}{N - \frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1}}.$$

In order to proceed we require an estimate of $\frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1}$. For $\alpha \in (0, 1)$ we have

$$\ln \left(\frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1} \right) = \ln(1-\alpha) + \sum_{i=2}^{N-1} \ln \frac{i-\alpha}{i} = \ln(1-\alpha) + G(\alpha) - G(0),$$

where we define $G(x) \triangleq \sum_{i=2}^{N-1} \ln(i-x)$. Now, assuming $x \in (0, 1)$, we have

$$G'(0) = - \sum_{i=2}^{N-1} \frac{1}{i} = -H(N-1) + 1, \text{ and } -\frac{\pi^2}{6} \leq - \sum_{i=2}^{N-1} \frac{1}{(i-x)^2} \leq G''(x) \leq 0,$$

and so we obtain

$$-\alpha(H(N-1) - 1) - \frac{\pi^2}{12} \alpha^2 \leq G(\alpha) - G(0) \leq -(H(N-1) - 1)\alpha.$$

Since $\ln(1 - \alpha) = -\alpha - \frac{1}{2}\alpha^2 - \dots$, we write

$$\ln\left(\frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1}\right) = -\alpha H(N-1) - c(N, \alpha)\alpha^2,$$

where, for $\alpha < \frac{1}{2}$, we have $0 \leq c(N, \alpha) \leq 2$. Choose $\alpha = \alpha(N)$ so that $\frac{N}{\alpha} \binom{\alpha}{N} (-1)^{N-1} = (1 - \epsilon)$. It follows that

$$r(N) = 1 - \epsilon + O(1/N), \quad \delta(N) \leq \frac{1 - (1 - \epsilon)}{N - (1 - \epsilon)} = O(1/N). \quad \diamond$$

EXAMPLE 3.90 (HEAVY-TAIL POISSON DISTRIBUTION). We get a second example if we start with

$$\begin{aligned} \hat{\lambda}_\alpha(x) &\triangleq -\frac{1}{\alpha} \ln(1 - x) = \frac{1}{\alpha} \sum_{i=1}^{\infty} \frac{x^i}{i}, \text{ and} \\ \rho_\alpha(x) &\triangleq e^{\alpha(x-1)} = e^{-\alpha} \sum_{i=0}^{\infty} \frac{\alpha^i x^i}{i!}. \end{aligned}$$

The detailed calculations for this case are left to Problem 3.19. \diamond

§3.16. GALLAGER'S LOWER BOUND ON THE DENSITY

We have seen in the previous section two sequences of capacity-achieving degree distribution pairs, and in fact there is no lack of other examples. Faced with such ample choice, which one is “best”? There are many possible and reasonable criteria upon which we could decide. E.g., we could investigate how quickly the finite-length behavior approaches the asymptotic limit for the various choices. This is done in Sections 3.22 and 3.23. Another important point of view is to introduce the notion of *complexity*. As we discussed in Section 3.5, on the BEC the decoding complexity is in one-to-one correspondence with the number of edges in the graph, since we use each edge at most once. How sparse can a graph be in order to achieve a fraction $(1 - \delta)$ of capacity? In the setting of LDPC ensembles under BP decoding Theorem 3.85 gives the answer. We will now see a more general information theoretic bound which applies to any (sequence of) code(s) and any decoding algorithm.

DEFINITION 3.91 (DENSITY OF PARITY-CHECK MATRICES). Let C be a binary linear code of length n and rate r and let H be a parity-check matrix of C . The *density* of H , denote it by $\Delta(H)$, is defined as

$$\Delta(H) \triangleq \frac{1}{nr} |\{(i, j) : H_{i,j} \neq 0\}|. \quad \nabla$$

EXAMPLE 3.92 (DENSITY OF STANDARD LDPC ENSEMBLE). The density of a degree distribution pair $(\Lambda, P) \triangleq (n, L, R) \triangleq (n, \lambda, \rho)$ is equal to

$$\Lambda'(1) \frac{1}{nr} = P'(1) \frac{1}{nr} = L'(1) \frac{1}{r} = R'(1) \frac{1-r}{r} = \frac{1}{\int \lambda} \frac{1}{r} = \frac{1}{\int \rho} \frac{1-r}{r}.$$

This density is equal to the decoding complexity of the BP decoder, when measured *per information bit*. \diamond

THEOREM 3.93 (LOWER BOUND ON DENSITY). Let $\{C_N\}$ be a sequence of binary linear codes in which codewords are used with uniform probability and which achieve a fraction $(1-\delta)$ of the capacity of the BEC(ϵ) with vanishing bit erasure probability. Let $\{\Delta_N\}$ be the corresponding sequence of densities of their parity-check matrices. Then,

$$(3.94) \quad \liminf_{N \rightarrow \infty} \Delta_N > \frac{K_1 + K_2 \ln \frac{1}{\delta}}{1 - \delta},$$

where

$$K_1 = \frac{\epsilon \ln \left(\frac{\epsilon}{1-\epsilon} \right)}{(1-\epsilon) \ln \left(\frac{1}{1-\epsilon} \right)}, \quad K_2 = \frac{\epsilon}{(1-\epsilon) \ln \left(\frac{1}{1-\epsilon} \right)}.$$

Proof. Consider a binary linear code $C[n, nr]$. Let X be chosen uniformly at random from $C[n, nr]$ and let Y be the received word after transmission over a BEC(ϵ). Let \mathcal{E} denote the random index set of erasure positions. There is a one-to-one correspondence between Y and $(Y_{\mathcal{E}}, \mathcal{E})$ as well as between (X, \mathcal{E}) and $(X_{\mathcal{E}}, X_{\mathcal{E}^c}, \mathcal{E})$. Further, $X_{\mathcal{E}^c} = Y_{\mathcal{E}^c}$. We have

$$\begin{aligned} H(X|Y) &= H(X|Y_{\mathcal{E}}, \mathcal{E}) = H(X, \mathcal{E}|Y_{\mathcal{E}}, \mathcal{E}) \\ &= H(X_{\mathcal{E}}, X_{\mathcal{E}^c}, \mathcal{E}|Y_{\mathcal{E}}, \mathcal{E}) = H(X_{\mathcal{E}}, X_{\mathcal{E}^c}|Y_{\mathcal{E}}, \mathcal{E}) = H(X_{\mathcal{E}}|Y_{\mathcal{E}}, \mathcal{E}) \\ &= \sum_{y_{\mathcal{E}}, E} H(X_{\mathcal{E}}|Y_{\mathcal{E}} = y_{\mathcal{E}}, \mathcal{E} = E) p(Y_{\mathcal{E}} = y_{\mathcal{E}}, \mathcal{E} = E) \\ &= \sum_{y_{\mathcal{E}}, E} (|E| - \text{rank}(H_E)) p(Y_{\mathcal{E}} = y_{\mathcal{E}}, \mathcal{E} = E) \\ &= \sum_E (|E| - \text{rank}(H_E)) p(\mathcal{E} = E) = n\epsilon - \sum_E \text{rank}(H_E) p(\mathcal{E} = E). \end{aligned}$$

The rank of H_E is upper bounded by the number of non-zero rows of H_E . This number in turn is equal to the number of parity-check nodes which involve erased bits. Therefore, $\sum_E \text{rank}(H_E) p(\mathcal{E} = E)$ is upper bounded by the average (where the average is over the channel realization) number of parity-checks which involve erased bits. Let $P(x)$ denote the check node degree distribution from a node perspective

(such a degree distribution is defined regardless whether the code is low-density or not) and let $R(x)$ denote the normalized such quantity, $R(x)n(1-r) = P(x)$. If a parity-check node is of degree i then the probability that it involves at least one erased bit is equal to $1 - (1-\epsilon)^i$, and therefore the average number of parity-check nodes which involve at least one erased bit is equal to

$$\sum_i P_i(1 - (1-\epsilon)^i) = n(1-r) - P(1-\epsilon) \leq n(1-r)(1 - (1-\epsilon)^{R'(1)}).$$

In the last step we have used Jensen's inequality (1.61). We therefore have

$$(3.95) \quad \frac{H(X|Y)}{n} \geq \epsilon - (1-r)(1 - (1-\epsilon)^{R'(1)}).$$

Assume now that we have a sequence of codes $\{C_N\}$ with asymptotic gap δ , i.e., $\delta_N \xrightarrow{N \rightarrow \infty} \delta$, and therefore $r_N \xrightarrow{N \rightarrow \infty} (1-\delta)(1-\epsilon)$. From Fano's inequality (1.49) we know that $\frac{1}{n_N}H(X|Y)$ must converge to zero in order for the bit erasure probability to tend to zero. We therefore have

$$\liminf_{N \rightarrow \infty} (1 - (1-\delta_N)(1-\epsilon))(1 - (1-\epsilon)^{R'_N(1)}) \geq \epsilon.$$

Solving this equation for $R'_N(1)$ we get

$$\liminf_{N \rightarrow \infty} R'_N(1) \geq \frac{\ln\left(1 + \frac{\epsilon}{\delta_N(1-\epsilon)}\right)}{\ln\left(\frac{1}{1-\epsilon}\right)} > \frac{\ln\left(\frac{\epsilon}{\delta_N(1-\epsilon)}\right)}{\ln\left(\frac{1}{1-\epsilon}\right)}.$$

To finish the proof observe that

$$\begin{aligned} \liminf_{N \rightarrow \infty} \Delta_N &= \liminf_{N \rightarrow \infty} R'_N(1) \frac{1-r_N}{r_N} = \liminf_{N \rightarrow \infty} R'_N(1) \frac{1 - (1-\delta_N)(1-\epsilon)}{(1-\delta_N)(1-\epsilon)} \\ &\geq \liminf_{N \rightarrow \infty} R'_N(1) \frac{\epsilon}{(1-\delta_N)(1-\epsilon)} > \frac{\ln\left(\frac{\epsilon}{\delta(1-\epsilon)}\right)}{\ln\left(\frac{1}{1-\epsilon}\right)} \frac{\epsilon}{(1-\delta)(1-\epsilon)} \\ &= \frac{K_1 + K_2 \ln \frac{1}{\delta}}{1-\delta}. \end{aligned} \quad \square$$

§3.17. OPTIMALLY-SPARSE DEGREE DISTRIBUTION PAIRS

Surprisingly, check-concentrated degree distribution pairs are essentially optimal with respect to a complexity versus performance trade-off. More precisely, the next theorem states that if we pick N and α for the check-concentrated ensemble carefully then the trade-off between gap δ and complexity Δ is up to a small additive constant the best possible. We skip the somewhat tedious proof.

THEOREM 3.96 (OPTIMALITY OF CHECK-CONCENTRATED DISTRIBUTION). Consider the check-concentrated ensembles introduced in Example 3.89 and transmission over the BEC(ϵ). Choose

$$N = \max \left(\left\lceil \frac{1 - c(\epsilon)(1 - \epsilon)(1 - \delta)}{\delta} \right\rceil, \lceil (1 - \epsilon)^{-\frac{1}{\epsilon}} \rceil \right), \quad \alpha = \frac{\ln \frac{1}{1 - \epsilon}}{\ln N},$$

where $c(\epsilon) = (1 - \epsilon)^{\frac{\pi^2}{6}} e^{(\frac{\pi^2}{6} - \gamma)\epsilon}$ and where γ is the Euler-Mascheroni constant, $\gamma \approx 0.5772$. This degree distribution pair achieves at least a fraction $1 - \delta$ of the channel capacity with vanishing bit erasure probability under BP decoding. Further, the density Δ is upper bounded by

$$\begin{aligned} \Delta &\leq \frac{K_1 + K_2 \log \frac{1}{\delta} + K(\epsilon, \delta)}{1 - \delta} \left(1 + \frac{1 - \epsilon}{\epsilon} \delta \right) \\ (3.97) \quad &\leq \frac{K_1 + K_2 \log \frac{1}{\delta}}{1 - \delta} + \kappa + O(\delta), \end{aligned}$$

where K_1 and K_2 are the constants of Theorem 3.93,

$$K(\epsilon, \delta) = \frac{\epsilon \ln \left(1 + \frac{1 - \epsilon}{\epsilon} (\delta c(\epsilon) + (1 - c(\epsilon))) \right)}{(1 - \epsilon) \ln \frac{1}{1 - \epsilon}},$$

and where $\kappa \triangleq \max_{\epsilon \in [0, 1]} K(\epsilon, 0) \approx 0.5407$. Comparing (3.97) with (3.94), we see that, up to a small constant κ , right-concentrated degree distribution pairs are optimally sparse.

From the above discussion one might get the impression that essentially no further improvement is possible in terms of the performance-complexity trade-off. This is true within the current framework. But we will see in Chapter 7 that, by allowing more complicated graphical models (in particular by introducing so-called *state* nodes), better trade-offs can be achieved.

§3.18. DEGREE DISTRIBUTIONS WITH GIVEN MAXIMUM DEGREE

In practice we are not only concerned with complexity (average degree) but also with the maximum degree since large degrees imply slow convergence of the finite-length performance to the asymptotic threshold (see Section 3.22). A glance at Theorem 3.96 shows that, although the average degree only grows like $\ln \frac{1}{\delta}$, the maximum grows exponentially faster, namely like $\frac{1}{\delta}$. Assume therefore that we have a given bound on the maximum variable node degree, call it 1_{\max} . Recall that a degree distribution pair (λ, ρ) has a threshold of at least ϵ if

$$\epsilon \lambda (1 - \rho(1 - x)) - x \leq 0, \quad x \in [0, 1].$$

Assume that we fix the check node degree distribution ρ and define the function

$$f(x, \lambda_2, \dots, \lambda_{1_{\max}}) \triangleq \epsilon \lambda(1 - \rho(1 - x)) - x = \epsilon \sum_{i \geq 2} \lambda_i (1 - \rho(1 - x))^{i-1} - x.$$

The function f is *linear* in the variables λ_i , $i = 2, \dots, 1_{\max}$ (note that $\lambda_1 = 0$), and from $r(\lambda, \rho) = 1 - \int \rho / \int \lambda$, we see that the rate is an increasing function in $\sum \lambda_i / i$ (for fixed ρ). Therefore, we can use the continuous linear program

$$(3.98) \quad \max_{\lambda} \left\{ \sum_{i \geq 2} \lambda_i / i \mid \lambda_i \geq 0; \sum_{i \geq 2} \lambda_i = 1; f \leq 0; x \in [0, 1] \right\}$$

to find the degree distribution λ which maximizes the rate for a given right-hand side ρ and a given threshold ϵ . In practice, we use a finite program. We can avoid numerical problems for x around zero caused by this discretization by incorporating the stability condition $\lambda_2 \leq 1/(\epsilon \rho'(1))$ explicitly into the linear program.

If we exchange the roles of the variable and check nodes and let y denote the fraction of the erased messages emitted at the check nodes, then an equivalent condition is

$$1 - y - \rho(1 - \epsilon \lambda(y)) \leq 0, \quad y \in [0, 1].$$

Proceeding in the same fashion as before, we see that we can optimize the check node degree distribution for a fixed variable degree distribution. In order to find a good degree distribution pair (λ, ρ) we can proceed as follows. Start with a given degree distribution pair and iterate between these two optimization problems. In practice an even simpler procedure suffices. It has been observed that *check-concentrated* degree distributions seem to achieve optimal performance. In this case $\rho(x)$ is completely specified by fixing the average check node degree r_{avg} :

$$\rho(x) = \frac{r(r + 1 - r_{\text{avg}})}{r_{\text{avg}}} x^{r-1} + \frac{r_{\text{avg}} - r(r + 1 - r_{\text{avg}})}{r_{\text{avg}}} x^r,$$

where $r = \lfloor r_{\text{avg}} \rfloor$. Now run the linear program (3.98) for various values of r_{avg} until the optimum solution has been found.

EXAMPLE 3.99 (COMPARISON: OPTIMUM VERSUS CHECK-CONCENTRATED). Assume that we choose $1_{\max} = 8$ and $r = 6$, and we want to obtain a design rate of one-half. Following the above procedure we find the pair

$$\lambda(x) = 0.409x + 0.202x^2 + 0.0768x^3 + 0.1971x^6 + 0.1151x^7, \quad \rho(x) = x^5,$$

which yields a rate of $r(\lambda, \rho) \approx 0.5004$. This degree distribution pair has a threshold of $\epsilon^{\text{BP}}(\lambda, \rho) \approx 0.4810$, which corresponds to a gap of $\delta \approx 0.0359$.

In comparison, a quick check shows that the check-concentrated degree distribution pair with equal complexity and comparable rate has parameters $\alpha = \frac{1}{5}$ and $N = 13$. We get

$$\begin{aligned}\lambda(x) &= 0.416x + 0.166x^2 + 0.1x^3 + 0.07x^4 + 0.053x^5 + 0.042x^6 + \\ &\quad 0.035x^7 + 0.03x^8 + 0.026x^9 + 0.023x^{10} + 0.02x^{11} + 0.0183x^{12}, \\ \rho(x) &= x^5.\end{aligned}$$

The design rate is $r(\lambda, \rho) \approx 0.499103$ and the threshold is $\epsilon^{\text{BP}}(\lambda, \rho) \approx 0.480896$ (it is determined by the stability condition), so that $\delta \approx 0.0385306$. The second degree distribution pair is slightly worse in all respects (higher maximum degree, smaller rate, smaller threshold). \diamond

§3.19. PEELING DECODER AND ORDER OF LIMITS

Density evolution computes the limit

$$(3.100) \quad \lim_{\ell \rightarrow \infty} \lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [\mathbf{P}_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)],$$

i.e., we determined the limiting performance of an ensemble under a *fixed* number of iterations as the blocklength tends to infinity and then let the number of iterations tend to infinity. What happens if the order of limits is exchanged, i.e., how does the limit

$$(3.101) \quad \lim_{n \rightarrow \infty} \lim_{\ell \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [\mathbf{P}_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)]$$

behave? This limit corresponds to the more typical operation in practice: for each fixed length the BP decoder continues until no further progress is achieved. We are interested in the limiting performance as the blocklength tends to infinity. In Section 3.22 we discuss the finite-length analysis of LDPC ensembles under BP decoding. We will see how we can compute the performance for a particular length (and an infinite number of iterations). The following example shows the typical behavior.

EXAMPLE 3.102 (CONVERGENCE TO THRESHOLD FOR LDPC (n, x^2, x^5)). Consider the ensemble $\text{LDPC}(n, x^2, x^5)$. Figure 3.103 shows $\mathbb{E}_{\text{LDPC}(n, x^2, x^5)} [\mathbf{P}_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell = \infty)]$ as a function of ϵ for $n = 2^i$, $i = 6, \dots, 20$. More precisely, we consider an *expurgated* ensemble as discussed in Section 3.24. In the limit of large blocklengths, the bit erasure probability converges to zero for $\epsilon < \epsilon^{\text{BP}} \approx 0.4294$ and to a non-zero constant for values above the threshold. In particular, the threshold for the limit (3.101) is the same as the threshold for the limit (3.100). \diamond

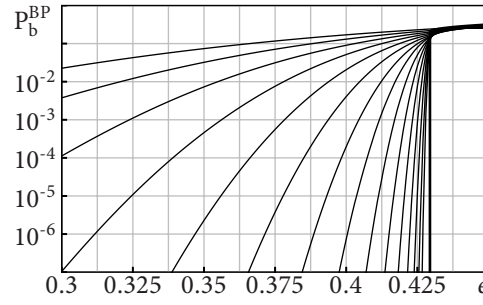


Figure 3.103: $\mathbb{E}_{\text{LDPC}(n, x^2, x^5)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell = \infty)]$ as a function of ϵ for $n = 2^i$, $i = 6, \dots, 20$. Also shown is the limit $\mathbb{E}_{\text{LDPC}(\infty, x^2, x^5)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell \rightarrow \infty)]$ which is discussed in Problem 3.17 (thick curve).

Motivated by this example, we will show that the thresholds corresponding to the two limits are always the same. Assume this for the moment and consider its consequence: it follows that the threshold is the same regardless of how the limit is taken (sequentially or jointly) as long as both n and ℓ tend to infinity. To see this, let $\tilde{\ell}(n)$ be any increasing function in n so that $\lim_{n \rightarrow \infty} \tilde{\ell}(n) = \infty$. Then, for any fixed ℓ

$$\lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \tilde{\ell}(n))] \leq \lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)],$$

because the error probability is a decreasing function in ℓ . Since this is true for any ℓ we can take the limit on the right to get

$$\lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \tilde{\ell}(n))] \leq \lim_{\ell \rightarrow \infty} \lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)].$$

On the other hand, using again the monotonicity with respect to ℓ , we have for a fixed n

$$\lim_{\ell \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)] \leq \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \tilde{\ell}(n))].$$

Taking the limit with respect to n we get

$$\lim_{n \rightarrow \infty} \lim_{\ell \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)] \leq \lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \tilde{\ell}(n))].$$

If we assume that

$$\lim_{n \rightarrow \infty} \lim_{\ell \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)] = \lim_{\ell \rightarrow \infty} \lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [P_b^{\text{BP}}(\mathbf{G}, \epsilon, \ell)],$$

then it follows from the above two inequalities that

$$\lim_{n \rightarrow \infty} \mathbb{E}_{\text{LDPC}(n, \lambda, \rho)} [\mathbb{P}_b^{\text{BP}}(\mathcal{G}, \epsilon, \tilde{\ell}(n))]]$$

also has the same limit.

The key to the analysis is the so-called *peeling* decoder. This decoder has identical performance to the message-passing decoder: if you take a sneak preview of Section 3.22 you will see that, unless we terminate the decoder prematurely (i.e., unless the number of iterations is too small), the message-passing decoder gets stuck in the largest “stopping set” which is contained in the set of erased bits. Such a stopping set is a subset of the variable nodes together with its outgoing edges so that no check node has induced degree one. From the description of the peeling decoder below you will see that the peeling decoder gets stuck in exactly the same structure. The performance of the two decoders is therefore identical (assuming an infinite number of iterations).

Although the two decoders have identical performance, the computation rules of the peeling decoder differ from those of the message-passing one in two aspects: (i) at the variable nodes we do not obey the message-passing principle but we replace the received value with the current estimate of the bit based on *all* incoming messages and the received message; (ii) rather than updating all messages in parallel we pick in each step one check node and update its outgoing messages as well as the messages of its neighboring variable nodes.

In addition we can apply the following simplifications without changing the behavior of the algorithm: once a non-erasure message has been sent out along a check node this check node has served its purpose and it no longer plays a role in the future of the decoding. This is true since a check node sends out a non-erased message only if all but possibly one of its neighbors are known. Therefore, after processing this check node *all* its neighbors are known. It follows from this observation that we can safely delete from the graph any such check node and all its attached edges. In the same manner, each known variable node can send to its neighboring check node its value and these values are accumulated at the check node. After that we can remove the known variable node and its outgoing edges from the graph. This procedure gives rise to a sequence of *residual* graphs. Successful decoding is equivalent to the condition that the sequence of residual graphs reaches the empty graph.

EXAMPLE 3.104 (PEELING DECODER APPLIED TO $[7, 4, 3]$ HAMMING CODE). Rather than giving a formal definition of the decoder, let us apply the peeling decoder to the $[7, 4, 3]$ Hamming code with received word $(1, 1, ?, ?, 1, 0, 0)$. This is shown in Figure 3.105. The top left-most picture shows the initial graph and the received word. The following two pictures are part of the initialization: (i) known variable nodes send their values along all outgoing edges; (ii) these values are accumulated at the

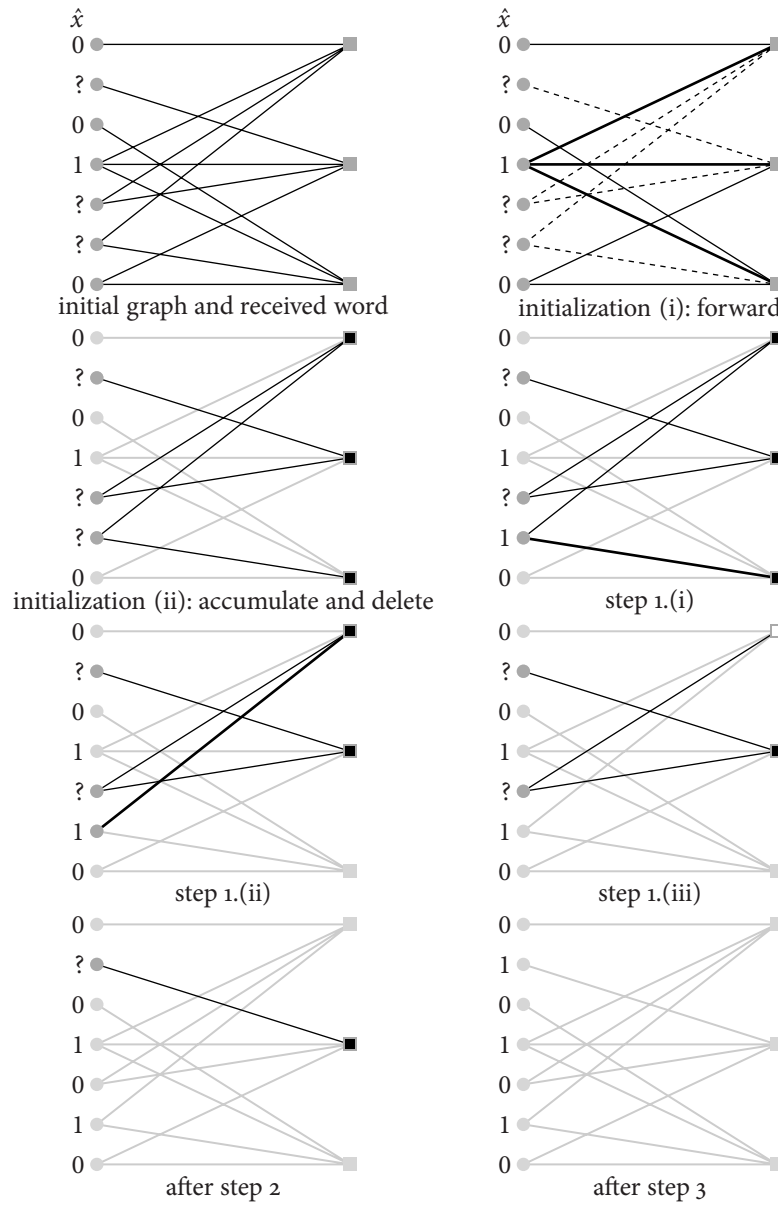


Figure 3.105: Peeling decoder applied to the $[7, 4, 3]$ Hamming code with the received word $y = (0, ?, ?, 1, 0, ?, 0)$. The vector \hat{x} indicates the current estimate of the decoder of the transmitted codeword x . After three decoding steps the peeling decoder has successfully recovered the codeword.

check nodes (black indicates an accumulated value of 1 and white an accumulated value of 0) and all known variable nodes and their connected edges are removed. After the initialization each decoding step consists of the following: (i) choose a check node of residual-degree one uniformly at random and forward the accumulated value to the connected variable node whose value is now determined; (ii) delete the chosen check node and its connected edge and forward the value of the newly determined variable node to all its remaining neighbors (check nodes); (iii) accumulate the forwarded values at the check nodes and delete the variable node and its connected edges.

In our example there is at each step only a single check node of residual-degree one. After three decoding steps the residual graph is the empty graph – the decoder has succeeded in determining the codeword. \diamond

We will now see that if we apply the peeling decoder to elements of an ensemble and if we increase the blocklength, then the sequence of residual graphs closely follows a “typical path.” We will describe this path, characterize the typical deviation from it, and relate this path to standard density evolution. As we discussed above, running the peeling decoder until it is stuck is equivalent to running the message-passing decoder for an infinite number of iterations. Therefore, if we can show that the peeling decoder has a threshold equal to the threshold ϵ^{BP} computed via density evolution, then we have in effect shown that the two limits (3.100) and (3.101) agree.

The proof of the following theorem is relegated to Section C.4.

THEOREM 3.106 (EVOLUTION OF RESIDUAL GRAPH FOR PEELING DECODER). Consider the performance of the ensemble LDPC (n, L, R) under the peeling decoder. Assume that the normalized degree distributions L and R have bounded degrees, call them l_{\max} and r_{\max} , respectively. Let t denote time, $t \in \mathbb{N}$. Time starts at zero and increases by one for every variable node which we peel off. Let $(L(t), R(t))$ denote the expected degree distribution pair at time t , where the normalization of $L(t)$ is with respect to n and the normalization of $R(t)$ is with respect to $n(1-r)$. More precisely, $nL_i(t)$ ($n(1-r)R_i(t)$) denotes the number of variable (check) nodes in the residual graph at time t .

Consider $\tilde{L}_i(y)$ and $\tilde{R}_i(y)$ given by

$$(3.107) \quad \begin{aligned} \tilde{L}_i(y) &= \epsilon L_i y^i, \quad i \geq 2, \\ \tilde{R}_0(y) &= 1 - \sum_{j \geq 1} \tilde{R}_j(y), \end{aligned}$$

$$(3.108) \quad \tilde{R}_1(y) = R'(1) \epsilon \lambda(y) [y - 1 + \rho(1 - \epsilon \lambda(y))],$$

$$(3.109) \quad \tilde{R}_i(y) = \sum_{j \geq 2} R_j \binom{j}{i} (\epsilon \lambda(y))^i (1 - \epsilon \lambda(y))^{j-i}, \quad i \geq 2.$$

If $\epsilon < \epsilon^{\text{BP}}$ then with probability at least $1 - O(n^{5/6} e^{-\frac{\sqrt{2n}}{1_{\max}}})$ the normalized degree distribution of the sequence of residual graphs of a specific instance deviates from the expected degree distribution by at most $O(n^{-1/6})$ uniformly from the start of the process until the total number of nodes in the residual graph has reached size ηn , where $\eta > 0$.

If $\epsilon > \epsilon^{\text{BP}}$ then with probability at least $1 - O(n^{5/6} e^{-\frac{\sqrt{2n}}{1_{\max}}})$ the normalized degree distribution of the sequence of residual graphs of a specific instance deviates from the expected degree distribution by at most $O(n^{-1/6})$ uniformly from the start of the process until the decoder gets stuck.

EXAMPLE 3.110 (EVOLUTION FOR (3, 6)-REGULAR ENSEMBLE). Assume that we apply the peeling decoder to the (3, 6)-regular ensemble. What is the evolution of the degree distribution of the residual graph? From the description of the algorithm, a variable is either removed or it retains all of its edges. Therefore, all remaining variable nodes are of degree 3 for our example. According to Theorem 3.106, the fraction of remaining variables (with respect to n) as a function of the parameter y is ϵy^3 . Here, $y = 1$ corresponds to the beginning of the process just after the initialization step. This initial fraction ϵ is easily verified: we expect a fraction ϵ of the original variable nodes to be erased by the channel so that a fraction ϵ of the variables is contained in the residual graph just after the initialization step.

The degree distribution on the check-node side is more interesting. Consider this degree distribution just after the initial step. Each edge of a check node is contained in the initial residual graph with probability ϵ . This means that e.g. the number of degree-one check nodes should be $\sum_j R_j \binom{j}{1} \epsilon^1 (1 - \epsilon)^{j-1}$. We can rewrite this as $R'(1) \epsilon \rho(1 - \epsilon)$, which agrees with the stated formula if we specialize it to $y = 1$. Figure 3.111 shows the evolution of the residual degrees $R_j(y)$, $j = 0, \dots, 6$. \diamond

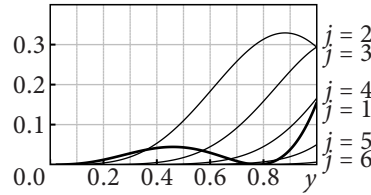


Figure 3.111: The evolution of the residual degrees $R_j(y)$, $j = 0, \dots, 6$, as a function of the parameter y for the (3, 6)-regular degree distribution. The channel parameter is $\epsilon = \epsilon^{\text{BP}} \approx 0.4294$. The curve corresponding to degree-one nodes is shown as a thick line.

From the stated expressions we can verify that the threshold under the peeling decoder is identical to the threshold under BP decoding. Consider the term $y -$