LDPC Codes with Local and Global Decoding

Eshed Ram       Yuval Cassuto
Andrew and Erna Viterbi Department of Electrical Engineering
Technion – Israel Institute of Technology, Haifa 32000, Israel
E-mails: {s6eshdr@campus, ycassuto@ee}.technion.ac.il

Abstract

This paper presents a theoretical study of a new type of LDPC codes that is motivated by practical storage applications. LDPC codes (suffix L represents locality) are LDPC codes that can be decoded either as usual over the full code block, or locally when a smaller sub-block is accessed (to reduce latency). LDPC codes are designed to maximize the error-correction performance vs. rate in the usual (global) mode, while at the same time providing a certain performance in the local mode. We develop a theoretical framework for the design of LDPC codes over the binary erasure channel. Our results include generalizing the density-evolution analysis to two dimensions, proving the existence of a decoding threshold and showing how to compute it, and constructing capacity-achieving sequences for any pair of local and global thresholds. In addition, a trade-off between the gap to capacity and the number of full-block accesses is studied, and a finite-length analysis of ML decoding is performed to exemplify a trade-off between the locality capability and the full-block error-correcting capability.

Keywords: Multi-block coding, codes with locality, two-sided Tanner graphs, binary erasure channel (BEC), capacity, density evolution (DE), iterative decoding, linear programming (LP), low-density parity-check (LDPC) codes.

I. INTRODUCTION

Low-density parity-check (LDPC) codes and their low-complexity iterative decoding algorithm [5] are a powerful method to achieve reliable communication and storage with rates that approach Shannon’s theoretical limit (see [12] and [16]). Due to their efficient encoding and decoding algorithms, communication applications such as WiFi, DVB, and Ethernet had adopted this
family of linear block codes. When used in data-storage applications, unlike in communications, retransmissions are not possible, and any decoding failure implies data loss; hence strong LDPC codes need to be provisioned for extreme data reliability. Another key feature of modern storage devices is fast access, i.e., low-latency and high-throughput read operations. However, high data reliability forces very large block sizes and high complexity, and thus degrades the device’s latency and throughput. This inherent conflict motivates a coding scheme that enables fast read access to small (sub) blocks with modest data protection and low complexity, while in case of failure providing a high data-protection "safety net" in the form of decoding a stronger code over a larger block. Our objective in this paper is to design LDPC codes to operate in such a multi-block coding scheme, where error-correction performance (vs. rate) is maximized in both the sub-block and full-block modes.

Formally, in a multi-block coding scheme a code block of length \( N \) is divided into \( M \) sub-blocks of length \( n \) (i.e., \( N = Mn \)). Each sub-block is a codeword of one code, and the concatenation of the \( M \) sub-blocks forms a codeword of another (stronger) code. This paper is the first to design LDPC codes for the multi-block scheme. Earlier work, such as [2] recently and [6], [7], [1] before, addressed the design of Reed-Solomon and related algebraic codes in multi-block schemes. While that prior work attests to the importance of the multi-block scheme, designing LDPC codes for it requires all-new tools and methods. Toward that, we define a new type of LDPC codes we call LDPC codes, where the suffix ‘L’ points to the code’s local access to its sub-blocks. The LDPC code is designed in such a way that each of the sub-blocks (of length \( n \)) can be decoded independently of the other sub-blocks (local decoding), and in addition the full block of length \( Mn \) can be decoded (global decoding) when local decoding fails. In order to fulfill this requirement, an LDPC code must have at least one parity-check matrix \( H \) in the form of

\[
H = \begin{pmatrix}
H_1 & 0 & \cdots & 0 & 0 \\
0 & H_2 & \cdots & 0 & 0 \\
\vdots & \ddots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & H_M \\
H_0
\end{pmatrix},
\]

where

\[
H_i \in \mathbb{F}_2^{m_i \times n}, \quad 1 \leq i \leq M,
\]
\[ H_0 \in \mathbb{F}_2^{m_0 \times M_0}. \]

Our theoretical results on analysis and construction of LDPCL codes lie upon the definition of the code through two distinct degree-distribution pairs. The \textit{local degree distribution} specifies the connections between sub-block variable nodes and their local check nodes, while the \textit{joint degree distribution} governs the connection of the global check nodes to variable nodes in the full block. The key challenge is to design the local and joint distributions such that both the local and the global (=local+joint composition) codes perform well. In particular, this requires the generalization of the binary erasure channel (BEC)’s density-evolution analysis \cite{15} to two dimensions (2D), where the local and joint dimensions are shown to have inherent asymmetries that need to be addressed to make the analysis work.

We note two prior works related to our results: \cite{10} proposed global coupling as a similar structure for LDPC codes, but were not interested in local-decoding performance and focused on algebraic structured codes over non-binary alphabets; \cite{14} studied a two-layer LDPC framework for a different application, but defined the codes through one big product degree distribution that is not amenable to asymptotic analysis and capacity-achieving constructions.

The paper is structured as follows. Section II reviews known results on LDPC codes that are essential to fully understand this paper, and presents the two-sided Tanner graph. Section III introduces the LDPCL ensembles, and Section IV details the asymptotic analysis of the belief-propagation (BP) decoding algorithm on these ensembles. In Section V we define a new notion of capacity-achieving sequences with both global and local decoding thresholds, prove that they exist, and provide a general construction for them. Section VI discusses a trade-off between the gap to capacity of an LDPCL ensemble to the number of decoding iterations that access the full block; minimizing access to full-block bits while decoding saves time and communication costs when the sub-blocks are distributed. In Section VII we perform a finite-length analysis for ML decoding of (regular) LDPCL codes, and finally, Section VIII summarizes the results and insights given in the paper.

II. Preliminaries

A. LDPC Codes: Review from \cite{15}

A linear block code is an LDPC code if it has at least one parity-check matrix that is sparse, i.e., the number of 1’s in \( H \) is linear in the block length. This sparsity enables the low-complexity (linear in block length) decoding algorithm. Every parity-check matrix \( H \) can be represented
by a bipartite graph, called a Tanner graph, with nodes partitioned to variable nodes and check nodes; there exists an edge between check node $i$ and variable node $j$, if and only if $H_{ij} = 1$ (this paper focuses on binary linear codes, but this representation can be generalized). The fraction of variable (resp. check) nodes in a Tanner graph with degree $i$ is denoted by $\Lambda_i$ (resp. $\Omega_i$), and the fraction of edges connected to variable (resp. check) nodes of degree $i$ is denoted by $\lambda_i$ (resp. $\rho_i$); $\Lambda_i$ and $\Omega_i$ are called node-perspective degree distributions, and $\lambda_i$ and $\rho_i$ are called edge-perspective degree distributions.

The degree-distribution polynomials associated to a Tanner graph are given by

$$\Lambda(x) = \sum_i \Lambda_i x^i, \quad \lambda(x) = \sum_i \lambda_i x^{i-1}, \quad x \in [0, 1],$$

(3a)

$$\Omega(x) = \sum_i \Omega_i x^i, \quad \rho(x) = \sum_i \rho_i x^{i-1}, \quad x \in [0, 1].$$

(3b)

The node-perspective and edge-perspective polynomials are related through

$$\Lambda(x) = \frac{\int_0^x \lambda(t) dt}{\int_0^1 \lambda(t) dt}, \quad \Omega(x) = \frac{\int_0^x \rho(t) dt}{\int_0^1 \rho(t) dt}, \quad x \in [0, 1],$$

(4a)

$$\lambda(x) = \frac{\Lambda'(x)}{\Lambda'(1)}, \quad \rho(x) = \frac{\Omega'(x)}{\Omega'(1)}, \quad x \in [0, 1],$$

(4b)

where the operator $'$ stands for the function’s derivative.

The rate of an LDPC code defined by a Tanner graph whose degree-distribution polynomials are given by (3a)–(3b) is lower bounded by the design rate, which is $1 - \frac{\Lambda'(1)}{\Omega'(1)} = 1 - \frac{\int_0^1 \rho(t) dt}{\int_0^1 \lambda(t) dt}$.

**B. The Two-Sided Tanner Graph**

We define an LDPCCL code of length $N = Mn$ through a two-sided Tanner graph. The main difference between the two-sided Tanner graph and the one-sided Tanner graph (i.e., the Tanner graph representing an ordinary LDPC code) is that in the former, the variable nodes are divided to $M$ disjoint sets of size $n$ each, and the check nodes are divided into two disjoint sets: local check nodes and joint check nodes. To distinguish between local and joint check nodes, the former are drawn to the right of the variable nodes and the latter are drawn to the left (hence its name: two-sided Tanner graph). The graph construction is constrained such that each local check node is connected only to variable nodes that are in the same sub-block of length $n$; the joint check-node connections have no constraints. The parity-check matrix corresponding to a two-sided Tanner graph is in the form of (1). The check nodes representing parity constraints in rows 1 to $\sum_{i=1}^M m_i$ are local check nodes, and the check nodes representing parity constraints in the last $m_0$ rows are joint check nodes. The set of edges in the graph is partitioned into two sets.
as well: edges connecting variable nodes to local check nodes (i.e., the right side of the graph) are called local edges and edges connecting variable nodes to joint check nodes are called joint edges. Finally, the local (resp. joint) degree of a variable node is the number of local (resp. joint) edges emanating from it.

**Example 1:** A two-sided Tanner graph and its corresponding parity-check matrix are given in Figure 1 and Figure 2 respectively: the parity check matrix is of size 12 × 18, \(n = 6\), \(M = 3\), and \(m_0 = m_1 = m_2 = m_3 = 3\).

![Fig. 1. A two-sided Tanner graph with \(n = 6\), \(M = 3\) and \(m_0 = m_1 = m_2 = m_3 = 3\).](image1)

![Fig. 2. The parity check matrix of the two-sided Tanner graph in Figure 1.](image2)

We denote by \(\Lambda_{L,i}\) the fraction of variable nodes with local degree \(i\), and by \(\Omega_{L,i}\) the fraction of local check nodes with degree \(i\). Similarly, \(\lambda_{L,i}\) designates the fraction of local edges connected to a variable node with local degree \(i\), and \(\rho_{L,i}\) designates the fraction of local edges connected to a local check node of degree \(i\). We call \((\Lambda_{L,i}, \Omega_{L,i}, \lambda_{L,i}, \rho_{L,i})\) local degree distributions. Note that we do not distinguish between local degree distributions of different sub-blocks, and we assume that they are the same in all sub-blocks (but the instances drawn from the distributions are in general different between the sub-blocks). The joint degree distributions \((\Lambda_{J,i}, \Omega_{J,i}, \lambda_{J,i}, \rho_{J,i})\) are defined similarly with an important difference. In contrast to Tanner graphs associated to ordinary LDPC codes where the minimal variable-node degree is usually 2 (i.e., \(\Lambda_{L,0} = \Lambda_{L,1} = 0\)), in the two-sided Tanner graph we allow some variable nodes to have joint degree 0 or 1. The reason for removing this restriction is related to the iterative decoding algorithm, and will become clearer in Sections IV and V. In the rest of the paper we will use \(P_0\) to denote the coefficient \(\Lambda_{J,0}\), due to its importance.

The local and joint degree-distribution polynomials \(\Lambda_L(\cdot), \lambda_L(\cdot), \Omega_L(\cdot), \rho_L(\cdot)\) and \(\Lambda_J(\cdot), \lambda_J(\cdot), \Omega_J(\cdot), \rho_J(\cdot)\)
are defined similarly to the degree-distribution polynomials for ordinary LDPC codes (3a)-(3b). Known relations between node-perspective and edge-perspective polynomials hold for the local polynomials. However, since some variable nodes may have a joint degree of zero, the equation that describes $\Lambda_J$ in terms of $\lambda_J$ changes to

$$\Lambda_J(x) = P_0 + (1 - P_0) \int_0^x \lambda_J(t) dt, \quad x \in [0, 1].$$

(5)

III. LDPC Ensembles

In this section we define the ensembles of two-sided Tanner graphs: the LDPCL ensembles. These ensembles have six parameters: $M, n, \Lambda_L(\cdot), \Lambda_J(\cdot), \Omega_L(\cdot),$ and $\Omega_J(\cdot)$. $M$ is the locality parameter that sets the number of sub-blocks in a code block, $n$ is the sub-block length, and $\Lambda_L(\cdot), \Lambda_J(\cdot), \Omega_L(\cdot), \Omega_J(\cdot)$ are the node-perspective degree-distribution polynomials; this ensemble is denoted by $LDPC(L, n, \Lambda_L, \Omega_L, \Lambda_J, \Omega_J)$. We can refer to an LDPCL ensemble through its edge-perspective degree-distribution polynomials, and then it is denoted by $LDPC(M, n, \lambda_L, \rho_L, \lambda_J, \rho_J, P_0)$ (when using the edge-perspective notation, one must specify $P_0$ as well).

A. Regular LDPC Ensembles

An LDPCL ensemble is called $(l_L, r_L, l_J, r_J)$-regular if in every two-sided Tanner graph in it, all of the variable nodes have fixed local and joint degrees of $l_L$ and $l_J$ (thus $P_0 = 0$), respectively, and all of the local and joint check nodes have fixed degrees of $r_L$ and $r_J$, respectively. The number of local and joint check nodes is therefore $Mn\frac{l_L}{r_L}$ and $Mn\frac{l_J}{r_J}$, respectively; hence, the design rate is

$$R = 1 - \frac{l_L}{r_L} - \frac{l_J}{r_J}.$$  

(6)

In what follows, we describe the sampling process from the $(l_L, r_L, l_J, r_J)$-regular ensemble. First, $M$ ordinary (not two-sided) Tanner graphs are independently sampled from the $(l_L, r_L)$-regular LDPC ensemble with block length $n$. These Tanner graphs are concatenated vertically with no inter-connections; call them local graphs. Another ordinary Tanner graph is then sampled from the ordinary $(l_J, r_J)$-regular LDPC ensemble with block length $N = Mn$. The latter graph is called the joint graph and it is flipped such that the check nodes are on the left side. Finally, the $Mn$ variable nodes of the joint graph are merged with the $Mn$ variable nodes of the $M$ local graphs to create a two-sided Tanner graph.
B. Irregular LDPCL Ensembles

An obvious generalization of the construction of regular LDPCL ensembles is the construction of irregular LDPCL ensembles in which the degrees of the nodes vary. As in ordinary LDPC codes, this generalization enables better iterative decoding performance. The sampling process from the irregular $LDPCL(M, n, \Lambda_L, \Lambda_J, \Omega_L, \Omega_J)$ ensemble is as follows. First, $M$ Tanner graphs are sampled independently from the $LDPC(n, \Lambda_L, \Omega_L)$ ensemble. These local graphs are concatenated vertically without inter-connections. Another Tanner graph is then sampled from the $LDPC(Mn, \Lambda_J, \Omega_J)$ ensemble. The latter joint graph is flipped, its $Mn$ variable nodes are randomly permuted, and merged with the $Mn$ variable nodes of the $M$ local graphs to create a two-sided Tanner graph. Note that the regular ensemble construction in Section III-A did not include the variable nodes random permutation step described above; we add it here to force statistical independence between the local and joint degrees of the variable nodes. The design rate of an $LDPCL(M, n, \Lambda_L, \Lambda_J, \Omega_L, \Omega_J)$ ensemble is given by

$$R(\Lambda_L, \Lambda_J, \Omega_L, \Omega_J) = 1 - \frac{\Lambda_L(1)}{\Omega_L'(1)} - \frac{\Lambda_J(1)}{\Omega_J'(1)} = 1 - \int_0^1 \frac{\rho_L(x)dx}{\lambda_L(x)dx} - \int_0^1 \frac{\rho_J(x)dx}{\lambda_J(x)dx} (1 - P_0). \quad (7)$$

We can see in (7), that setting $P_0 > 0$ allows increasing the code rate, which we later indeed do in our constructions.

IV. Iterative Decoding Asymptotic Analysis

A standard decoding algorithm for ordinary LDPC codes over the BEC is a message-passing algorithm known as the belief-propagation (BP) algorithm; its decoding complexity is linear in the block length and despite being sub-optimal, there are LDPC ensembles that approach the BEC capacity with BP. To take advantage of the locality structure of the Tanner graphs described above, the suggested decoding algorithm will operate in two modes: local mode and global mode. In the local mode, the decoder tries to decode a sub-block of length $n$ using BP on the local Tanner graph corresponding to the desired sub-block. If the decoder meets a failure criterion (e.g., maximum number of iterations or stable non-zero fraction of erased variable nodes), then it enters the global mode where it tries to decode the entire code block (of length $N = Mn$) using BP on the complete two-sided Tanner graph. If the decoder succeeds, it extracts the wanted sub-block, and if it fails, it declares a decoding failure.
The message scheduling in the global mode is a flooding schedule: in the first step of a
global decoding iteration, the variable nodes send messages to the local and joint check nodes
in parallel, and in the second step the local and global check nodes send their messages back
to the variable nodes. As in the case of ordinary LDPC codes, the messages passed are either
erasures or decoded 0/1 bits (later in Section VI we change the schedule from flooding to be
more locality aware).

Example 2: Figure 3 exemplifies the decoding algorithm for an LDPCCL code with locality
\( M = 2 \) and sub-block length of \( n = 12 \); the desired sub-block is the upper one. First, the
decoder tries to locally decode the upper sub-block using only its local parity checks (a). It fails
and enters the global mode (b), where it runs BP on the complete graph. In every iteration,
messages are sent in parallel to the local and joint side (c)-(d), until the decoder resolves all of
the variable nodes, and in particular, the desired sub-block.

In the local mode, the asymptotic (as \( n \to \infty \)) analysis of decoding algorithm is identical to
the asymptotic analysis of ordinary LDPC codes. Specifically, in the limit where \( n \to \infty \), there
exists a local decoding threshold denoted by \( \epsilon^*_L \) that depends on the degree distributions, such
that if the fraction of erasures \( \epsilon \) is less than \( \epsilon^*_L \), the decoder will resolve the desired sub-block.

Fig. 3. An example for local and global decoding where the desired sub-block is the upper one: (a) a complete
two-sided Tanner graph with erased (black) variable nodes before decoding; (b) the decoder fails to locally decode
the desired sub-block and gets stuck, then it enters the global mode; (c) after one iteration in global mode; (d) after
two iterations in global mode; (e) the decoder resolves the desired sub-block.

In the local mode, the asymptotic (as \( n \to \infty \)) analysis of decoding algorithm is identical to
the asymptotic analysis of ordinary LDPC codes. Specifically, in the limit where \( n \to \infty \), there
exists a local decoding threshold denoted by \( \epsilon^*_L \) that depends on the degree distributions, such
that if the fraction of erasures \( \epsilon \) is less than \( \epsilon^*_L \), the decoder will resolve the desired sub-block.
in the local mode with probability converging to 1, and if $\epsilon > \epsilon^*_L$, the decoder will fail in the local mode with probability converging to 1. Moreover, $\epsilon^*_L$ can be calculated numerically via

$$\epsilon^*_L = \inf_{(0,1]} \frac{x}{\lambda_L(1 - \rho_L(1 - x))}.$$  \hspace{1cm} (8)

In contrast to the local mode, in the global mode there is a major difference in the asymptotic analysis of the BP algorithm between LDPC and LDPCL codes. Due to the built-in structure of the LDPCL ensemble, i.e., the constraints on edge connections (see Section III), one cannot calculate the global degree-distribution polynomials (i.e., considering an ”effective” one-sided Tanner graph ensemble), and use them to find the global BP decoding threshold with an equation similar to (8) (except for some special cases detailed below).

### A. Decoding in Global Mode

In this sub-section, we analyze the asymptotic performance (as $n \to \infty$) of the BP decoding algorithm over the BEC in the global mode, and derive density-evolution equations for the LDPCL. Two important theorems about ordinary LDPC codes hold for LDPCL codes as well, namely, the concentration theorem and the convergence to tree channel theorem (see [15, Theorem 3.30, Theorem 3.50]). The former theorem allows us to analyze the performance of a particular LDPCL code via the analysis of the (expectation of the) ensemble as $n \to \infty$, and the latter theorem enables us to consider the tree ensembles [15, Chapter 3.7.2] corresponding to the general-graph ensembles of Section III as $n \to \infty$. We adopt the notations from [15] in the following derivation.

**Definition 1 (LDPCL Tree Ensemble):** Let $l \in \mathbb{N}$ and $\Lambda_L, \Omega_L, \Lambda_J, \Omega_J$ be degree-distribution polynomials. The tree ensemble $T_l(\Lambda_L, \Omega_L, \Lambda_J, \Omega_J)$ is constructed in the following way. Let $V(i,j)$ be a variable-node-rooted tree of depth 1, with $i$ local check nodes and $j$ joint check nodes as children. Let $C(i)$ be a check-node-rooted tree of depth 1 with $i$ variable-node children. $T_0(\Lambda_L, \Omega_L, \Lambda_J, \Omega_J)$ is the ensemble consisting of a single element which is a variable node. To sample from $T_l(\Lambda_L, \Omega_L, \Lambda_J, \Omega_J)$, $l \geq 1$, start by sampling from $T_{l-1}(\Lambda_L, \Omega_L, \Lambda_J, \Omega_J)$, and for each leaf variable node $v$, if $v$ is the root (i.e., $l = 1$) or $v$’s father is a local check node, replace it with $V(i,j)$ with probability $\lambda_{L,i+1} \cdot \Lambda_{J,j}$, and if $v$’s father is a joint check node replace it with $V(i,j)$ with probability $\Lambda_{L,i} \cdot \lambda_{J,j+1}$. Subsequently, replace each local check-node leaf with $C(i)$ with probability $\rho_{L,i+1}$ and each joint check-node leaf with $C(i)$ with probability $\rho_{J,i+1}$.

Figures 4 and 5 illustrate the $(l_L, r_L, l_J, r_J) = (2, 4, 1, 3)$-regular and the $(\Lambda_L(x) = x^2, \Omega_L(x) = x^3, \Lambda_J(x) = \frac{3}{4} + \frac{1}{4}x, \Omega_J(x) = x^2)$-irregular tree ensembles of depth 2, respectively. In the regular
Fig. 4. The single element of the $\mathcal{T}_2(\Lambda_L(x) = x^2, \Omega_L(x) = x^4, \Lambda_J(x) = x, \Omega_J(x) = x^3)$ (regular) ensemble. The local (resp. joint) check nodes are marked with ‘L’ (resp. ‘J’).

Fig. 5. The 8 elements of the $\mathcal{T}_2(\Lambda_L(x) = x^2, \Omega_L(x) = x^4, \Lambda_J(x) = x, \Omega_J(x) = x^3)$ (irregular) ensemble. The probability of each element is given above it. The local (joint) check nodes are marked with ‘L’ (‘J’).

ensemble there is only one element, and in the irregular ensemble there are 8 elements; the graphs are illustrated with their corresponding sampling probabilities. In both figures, the local check nodes are tagged with ‘L’ and the joint check nodes are tagged with ‘J’.

**Theorem 1 (Two-dimensional density evolution):** Consider a random element from the $LDPCL(M, n, \Lambda_L, \Lambda_J, \Omega_L, \Omega_J)$ ensemble. Let $x_l(\epsilon)$ and $y_l(\epsilon)$ denote the probability that a local and joint edge, respectively, carries a variable-to-check erasure message after $l$ BP iterations over
the BEC(ε) as n → ∞. Then,
\[
x_l(ε) = ε \cdot Λ_L (1 - ρ_L (1 - x_{l-1}(ε))) \cdot Λ_J (1 - ρ_J (1 - y_{l-1}(ε))), \quad l \geq 0, \quad (9a)
\]
\[
y_l(ε) = ε \cdot Λ_L (1 - ρ_L (1 - x_{l-1}(ε))) \cdot Λ_J (1 - ρ_J (1 - y_{l-1}(ε))), \quad l \geq 0, \quad (9b)
\]
\[
x_{-1}(ε) = y_{-1}(ε) = 1. \quad (9c)
\]

Proof: The convergence to tree channel theorem implies that as n → ∞, we can consider tree ensembles (see Definition 1) instead of the ensembles in Section III for finding the bit-error probability of the BP decoding algorithm. Consider first an \((l_L, l_J, r_L, r_J)\)-regular tree ensemble. The degree-distribution polynomials are \(Λ_L(x) = x^{l_L}, \lambda_L(x) = x^{l_L-1}, \rho_L(x) = x^{r_L-1}, \Lambda_J(x) = x^{l_J}, \lambda_J(x) = x^{l_J-1}, \rho_J(x) = x^{r_J-1}\). For every \(l \geq 0\), the tree ensemble of a \((l_L, l_J, r_L, r_J)\)-regular LDPCL ensemble consists of only one element which is a bipartite tree of depth \(2l\) rooted in a variable node, such that every local (resp. joint) check node has \(r_L-1\) (resp. \(r_J-1\)) children, every variable node which has a local check node as a father, including the root, has \(l_L-1\) local check-node children and \(l_J\) joint check-node children, and every variable node which has a joint check node as a father, has \(l_J-1\) joint check-node children and \(l_L\) local check-node children (see Figure 4).

We use mathematical induction on the number of iterations \(l\). For \(l = 0\) the probability that a local or joint edge carries an erasure as a variable-to-check message is \(ε\). Since \(x_{-1}(ε) = y_{-1}(ε) = 1\), then (9a)-(9b) with \(l = 0\) leads to
\[
x_0(ε) = ε \cdot \left(1 - (1 - 1)^{r_L-1}\right)^{l_L} \cdot \left(1 - (1 - 1)^{r_J-1}\right)^{l_J} = ε,
\]
\[
y_0(ε) = ε \cdot \left(1 - (1 - 1)^{r_L-1}\right)^{l_L} \cdot \left(1 - (1 - 1)^{r_J-1}\right)^{l_J-1} = ε,
\]
hence, (9a)-(9b) hold for regular ensembles and \(l = 0\). Assume correctness of (9a)-(9b) for some \(l \geq 0\), and consider iteration \(l + 1\). Each local (resp. joint) check node has \(r_L-1\) (resp. \(r_J-1\)) children variable nodes. Recall that in the BP decoding algorithm for the BEC, a check node will not send an erasure message to its father if and only if all of its incoming messages from its children are not erasures. Let \(u_l(ε)\) and \(w_l(ε)\) designate the probability that an outgoing message from a local and joint check node, respectively, is an erasure. Then,
\[
u_{l+1}(ε) = 1 - (1 - x_l(ε))^{r_L-1},
\]
\[
w_{l+1}(ε) = 1 - (1 - y_l(ε))^{r_J-1}. \quad (10)
\]
In addition, a variable-to-check message will be an erasure if and only if all of its incoming messages (including the one from the channel) are erasures. Since every variable node which
has a local (resp. joint) check node as a father, has \( l_L - 1 \) (resp. \( l_L \)) local check-node children and \( l_J \) (resp. \( l_J - 1 \)) joint check-node children,

\[
x_{l+1}(\epsilon) = \epsilon \cdot (w_{l+1}(\epsilon))^{l_L-1} \cdot (w_{l+1}(\epsilon))^{l_J},
\]

\[
y_{l+1}(\epsilon) = \epsilon \cdot (w_{l+1}(\epsilon))^{l_L} \cdot (w_{l+1}(\epsilon))^{l_J-1}.
\]

Combining (10) and (11) yields that (9a)-(9b) hold in the regular case for \( l + 1 \), thus by mathematical induction it holds for every \( l \geq 0 \).

The irregular case follows by invoking the convergence theorem and taking the expected value in the induction step (i.e., in (10) and (11)) w.r.t. the tree ensemble construction in Definition 1. Since the probability that a local (resp. joint) edge is connected to a local (resp. joint) check node of degree \( i \) is given by \( \rho_{L,i} \) (resp. \( \rho_{J,i} \)), and the probability that a local (resp. joint) edge is connected to a variable node of local degree \( i \) and joint degree \( j \) is given by \( \lambda_{L,i} \Lambda_{J,j} \) (resp. \( \Lambda_{L,i} \lambda_{J,j} \)), then

\[
u_{l+1}(\epsilon) = 1 - \rho_L \left(1 - x_l(\epsilon)\right),
\]

\[
u_{l+1}(\epsilon) = 1 - \rho_J \left(1 - y_l(\epsilon)\right),
\]

\[
x_{l+1}(\epsilon) = \epsilon \cdot \lambda_L (u_{l+1}(\epsilon)) \cdot \Lambda_J (w_{l+1}(\epsilon)),
\]

\[
y_{l+1}(\epsilon) = \epsilon \cdot \Lambda_L (u_{l+1}(\epsilon)) \cdot \lambda_J (w_{l+1}(\epsilon)) .
\]

We call (9a)-(9b) the 2D density-evolution (2D-DE) equations of LDPCL codes over the \textit{BEC}(\epsilon). These equations prove that the analysis of the BP decoding algorithm in the global mode differs from the analysis of the standard BP decoding LDPC codes over the \textit{BEC}(\epsilon). Although for some special cases the 2D-DE equations can be reduced to the well-known 1D-DE equations (Corollary 1 to follow), for the more interesting cases we will need the 2D-DE equations in (9a)-(9b). To simplify notations, \( \epsilon \) will be omitted from now on from \( x_l(\epsilon) \) and \( y_l(\epsilon) \) if it is clear from the context.

\textbf{Remark 1:} Although \( x_l \) and \( y_l \) in (9a)-(9b) seem symmetric to each other, it is not necessarily true since we allow variable nodes to have joint degrees 0 (\( P_0 > 0 \)) or 1 (\( \lambda_J(0) > 0 \)), while their local degrees are forced to be greater than 1. This asymmetry has a crucial effect on the global decoding process which is explained and detailed in Section IV-B.

\textbf{Corollary 1:} If for every \( x \in [0,1] \), \( \rho_L(x) = \rho_J(x) \), and the variable local and joint degrees are regular, i.e., \( \lambda_L(x) = x^{l_L-1}, \lambda_J(x) = x^{l_J-1} \), then

\[
x_l = y_l, \quad \forall l \geq 0,
\]

(12)
and

\[ x_l = \epsilon \lambda(1 - \rho(1 - x_{l-1})) \quad \forall l \geq 0, \quad (13) \]

where, \( \rho(x) \triangleq \rho_L(x), \lambda(x) \triangleq x^{L_l + J_l-1} \).

**Proof:** Since in the regular case we have

\[ \Lambda_L(x) = x \lambda_L(x), \quad x \in [0, 1], \]
\[ \Lambda_J(x) = x \lambda_J(x), \quad x \in [0, 1], \]

(12) and (13) follow immediately from (9a)-(9c).

Corollary \[ \text{[1]} \] asserts that if we use identical degree-distributions for local and joint check nodes and we force all variable nodes to have a local and joint regular degree, then the 2D-DE equations in (9a)-(9b) degenerate to the already known 1D-DE equation; hence, in this case we can use the analysis of BP decoding in ordinary LDPC codes for the LDPCL codes as well.

**Example 3:** Consider the \((l_L, l_J, r_L, r_J)\)-regular ensemble with \(l_L = 2, \ l_J = 1, \ r_L = r_J = 6\).

In view of (6), the corresponding design rate is

\[ R = \frac{1}{2}. \]

In addition,

\[ \lambda(x) = x^{l_L + l_J - 1} = x^2, \]
\[ \rho(x) = \rho_L(x) = x^5, \]

and due to Corollary \[ \text{[1]} \] we have

\[ x_l = \epsilon \lambda(1 - \rho(1 - x_{l-1})) \]
\[ = \epsilon(1 - (1 - x_{l-1})^5)^2 \]

which is the 1D-DE equation for the \((3, 6)\)-regular LDPC ensemble; in other words, in the local mode a \((2, 6)\)-regular LDPC code is decoded, and in the global mode a \((3, 6)\)-regular LDPC code is decoded. The local and global thresholds are therefore,

\[ \epsilon^*_L = 0.2, \quad \epsilon^*_G = 0.4294, \]

respectively.
B. Threshold

In this sub-section, the asymptotic global threshold of the BP decoding algorithm on LDPCL ensembles is studied. We prove that for any given LDPCL\((M,n,\lambda_L,\rho_L,\lambda_J,\rho_J,P_0)\) ensemble, there exists a threshold denoted by \(\epsilon^*_G\) such that when \(n \to \infty\), the BP algorithm will successfully globally decode (for sufficiently large number of iterations) a code block transmitted on the \(BEC(\epsilon)\) if and only if \(\epsilon < \epsilon^*_G\). A method to calculate this threshold is provided as well.

Define

\[
\begin{align*}
    f(\epsilon, x, y) &= \epsilon \lambda_L(1 - \rho_L(1 - x)) \Lambda_J(1 - \rho_J(1 - y)), \quad x, y, \epsilon \in [0, 1] \\
    g(\epsilon, x, y) &= \epsilon \Lambda_L(1 - \rho_L(1 - x)) \lambda_J(1 - \rho_J(1 - y)), \quad x, y, \epsilon \in [0, 1]
\end{align*}
\]  

such that (14a) and (14b) can be re-written as

\[
\begin{align*}
    x_l &= f(\epsilon, x_{l-1}, y_{l-1}), \quad l \geq 0 \\
    y_l &= g(\epsilon, x_{l-1}, y_{l-1}), \quad l \geq 0 \\
    x_{-1} &= y_{-1} = 1.
\end{align*}
\]  

Lemma 1: The functions \(f\) and \(g\) are monotonically non-decreasing in all of their variables.

Proof: Since the images of \(\lambda_L(\cdot), \Lambda_L(\cdot), \lambda_J(\cdot), \Lambda_J(\cdot), \rho_L(\cdot)\) and \(\rho_J(\cdot)\) lie in \([0, 1]\), then \(f\) and \(g\) are monotonically non-decreasing in \(\epsilon \in [0, 1]\). The rest of the proof is similar and is left as an exercise.

Definition 2: Let \(\epsilon \in (0, 1)\). We say that \((x, y) \in [0, 1]^2\) is an \((f, g)\)-fixed point if

\[
\begin{pmatrix}
x \\
y
\end{pmatrix} = \begin{pmatrix}
f(\epsilon, x, y) \\
g(\epsilon, x, y)
\end{pmatrix}.
\]  

Clearly, for every \(\epsilon \in (0, 1), (x, y) = (0, 0)\) is a trivial \((f, g)\)-fixed point. However, it is not clear yet if there exists a non-trivial \((f, g)\)-fixed point. In particular, we ask: for which choices of \(\epsilon \in (0, 1), \lambda_L, \rho_L, \lambda_J, \rho_J\) and \(P_0\) there exists a non-trivial \((f, g)\)-fixed point? The following lemmas will help answering this question.

Lemma 2: Let \(\epsilon \in (0, 1)\), and let \((x, y) \in [0, 1]^2\) be an \((f, g)\)-fixed point. Then,

1) \(x = 0\) implies \(y = 0\), and if \(P_0 = 0\) or \(\lambda_J(0) > 0\), then \(y = 0\) implies \(x = 0\).

2) \((x, y) \in [0, \epsilon)^2\).

3) If \(\{x_l\}_{l=0}^\infty\) and \(\{y_l\}_{l=0}^\infty\) are defined by (15), then

\[
x_l \geq x, \quad y_l \geq y, \quad \forall l \geq 0.
\]  

(17)
Proof: See Appendix A.

Remark 2: Item 1) in Lemma 2 expresses the asymmetry between the local and joint sides during the decoding algorithm discussed in Remark 1.

Lemma 3: Let \( x_l \) and \( y_l \) be defined by (15) and let \( 0 < \epsilon \leq \epsilon' < 1 \). Then,
\[
x_{l+1}(\epsilon) \leq x_l(\epsilon), \quad \forall l \geq 0
\]
and
\[
y_{l+1}(\epsilon) \leq y_l(\epsilon), \quad \forall l \geq 0,
\]
and
\[
x_l(\epsilon) \leq x_l(\epsilon'), \quad \forall l \geq 0
\]
\[
y_l(\epsilon) \leq y_l(\epsilon'), \quad \forall l \geq 0.
\]

Proof: By mathematical induction on \( l \) and by Lemma 1. The details are left as an exercise.

In view of (15), it can be verified that
\[
x_l(0) = 0, \quad x_l(1) = 1, \forall l \geq 0
\]
\[
y_l(0) = 0, \quad y_l(1) = 1, \forall l \geq 0.
\]
Since \( x_l \) and \( y_l \) are bounded from below by 0, then Lemma 3 implies that the limits \( \lim_{l \to \infty} x_l(\epsilon) \) and \( \lim_{l \to \infty} y_l(\epsilon) \) exist. In view of (20) we can define a BP global decoding asymptotic threshold by
\[
\epsilon^*_G = \sup \left\{ \epsilon \in [0,1]: \lim_{l \to \infty} y_l(\epsilon) = \lim_{l \to \infty} x_l(\epsilon) = 0 \right\}.
\]
Note that from continuity of \( g \) in (14b), Item 1) in Lemma 2 implies that if \( \lim_{l \to \infty} x_l(\epsilon) = 0 \), then \( \lim_{l \to \infty} y_l(\epsilon) = 0 \). Thus, (21) can be re-written as
\[
\epsilon^*_G = \sup \left\{ \epsilon \in [0,1]: \lim_{l \to \infty} x_l(\epsilon) = 0 \right\}.
\]

Theorem 2 (Fixed-point characterization): Let
\[
\hat{\epsilon} = \sup \{ \epsilon \in [0,1]: (16) \text{ has no solution with } (x,y) \in (0,1] \times [0,1] \}.
\]
Then, \( \epsilon^*_G = \hat{\epsilon} \).

Proof: Let \( \epsilon < \hat{\epsilon} \), and let \( x(\epsilon) = \lim_{l \to \infty} x_l(\epsilon), \ y(\epsilon) = \lim_{l \to \infty} y_l(\epsilon) \). Taking the limit \( l \to \infty \) in (15) yields that \( (x(\epsilon), y(\epsilon)) \) is an \((f,g)\)-fixed point. In view of (23), since \( \epsilon < \hat{\epsilon} \), it follows that \( x(\epsilon) = 0 \). From (22) we have \( \epsilon < \epsilon^*_G \), for every \( \epsilon < \hat{\epsilon} \); this implies that \( \hat{\epsilon} \leq \epsilon^*_G \).
For the other direction let $\epsilon > \hat{\epsilon}$ and let $(z_1, z_2)$ be an $(f,g)$-fixed point such that $z_1 > 0$. Lemma 2-Item 3) implies that $x_l(\epsilon) \geq z_1 > 0$, $\forall l \geq 0$, thus $\lim \limits_{l \to \infty} x_l(\epsilon) > 0$, where the existence of this limit is assured due to Lemma 3; hence, $\epsilon > \epsilon^*_G$.

Since this is true for all $\epsilon > \hat{\epsilon}$, then we deduce that $\hat{\epsilon} \geq \epsilon^*_G$ and complete the proof.

We proceed by providing a numerical way to calculate the threshold of a given choice of $\Lambda_L, \Lambda_J, \Omega_L$ and $\Omega_J$. Define

$$q_L(x) \triangleq x \cdot \frac{\Lambda_L (1 - \rho_L (1 - x))}{\lambda_L (1 - \rho_L (1 - x))}, \quad x \in (0, 1]$$

$$q_J(x) \triangleq x \cdot \frac{\Lambda_J (1 - \rho_J (1 - x))}{\lambda_J (1 - \rho_J (1 - x))}, \quad x \in (0, 1]$$

(24)

**Lemma 4:** $\lim \limits_{x \to 0} q_L(x) = 0$.

**Proof:** See Appendix B.

Since $q_L(1) = 1$, Lemma 4 and the intermediate-value theorem imply that for every $w \in (0, 1]$ there exists $x \in (0, 1]$ such that $q_L(x) = w$; it is not true in general that $\lim \limits_{x \to 0} q_J(x) = 0$, and this limit may be infinite (check for example the case $P_0 > 0$, $\rho_J(x) = x^3$ and $\lambda_J(x) = x^2$).

**Definition 3:** For every $y > 0$ such that $q_J(y) \leq 1$ define

$$q(y) \triangleq \max \{ x : q_L(x) = q_J(y) \}.$$  

(25)

**Lemma 5:** If $(x, y)$ is an $(f,g)$-fixed point with $y > 0$, then

$$x \leq q(y).$$

(26)

**Proof:** See Appendix C.

**Remark 3:** If $q_L(\cdot)$ is injective, then (25) and (26) can be restated as

$$x = q(y) = q_L^{-1}(q_J(y)).$$

(27)

**Theorem 3 (Threshold numerical calculation):** Let $\lambda_L, \rho_L, \lambda_J, \rho_J$ be degree-distribution polynomials, let $P_0 \in [0, 1]$, and let $\epsilon^*_G = \epsilon^*_G(\Lambda_L, \Lambda_J, \Omega_L, \Omega_J)$ be the BP global decoding threshold of the LDPCL($M, n, \Lambda_L, \Lambda_J, \Omega_L, \Omega_J$) ensemble on the BEC when $n \to \infty$.

If $P_0 = 0$ or $\lambda_J(0) > 0$, then

$$\epsilon^*_G = \inf \limits_{y \in [0, 1]} \frac{y}{q_J(1, q(y), y)}.$$  

(27)
Else,
\[
\epsilon^*_G = \min \left\{ \inf_{y \in (0,1]} \frac{y}{g(1,q(y),y)}, \frac{1}{P_0} \cdot \inf_{(0,1]} \frac{x}{\lambda_L(1 - \rho_L(1 - x))} \right\}.
\] (28)

Proof: See Appendix D. □

Remark 4: In the special case where \(\rho_J(\cdot) = \rho_L(\cdot)\), and the variable local and joint degrees are regular (i.e. \(\lambda_L(x) = x^{1-\epsilon}, \lambda_J(x) = x^{1-\epsilon}\)), (24) implies that \(q_L(\cdot)\) is injective, and that \(q_L(x) = q_J(x)\), for every \(x \in (0,1]\). Therefore, in view of Remark 3, every \((f,g)\)-fixed point lies on the line \(y = x\). Further, since \(P_0 = 0\), Theorem 3 yields
\[
\epsilon^*_G = \inf_{y \in (0,1]} \frac{y}{g(1,q(y),y)} \\
= \inf_{y \in (0,1]} \frac{y}{g(1,y,y)} \\
= \inf_{y \in (0,1]} \frac{y}{\lambda(1 - \rho(1 - y))},
\]
where for every \(y \in [0,1]\), \(\rho(y) = \rho_L(y)\), and \(\lambda(y) = y^{1+1-1}\). This is consistent with Corollary 1.

Example 4: Consider an LDPCL ensemble characterized by
\[
\lambda_L(x) = x, \quad \rho_L(x) = x^9 \\
\lambda_J(x) = 0.3396x + 0.6604x^4, \quad P_0 = 0.2667, \quad \rho_J(x) = x^9.
\]
Using (7), (8) and (28), the design rate is \(R = 0.5571\), the local decoding threshold is \(\epsilon^*_L = 0.1112\), and the global decoding threshold is \(\epsilon^*_G = 0.35\). Figure 6 illustrates the 2D-DE equations in (9a)-(9c) for three different bit erasure probabilities: 0.33, 0.35, 0.37, from left to right, respectively. When the channel’s erasure probability is \(\epsilon = 0.33\), there are no \((f,g)\)-fixed points – the decoding process ends successfully, and when \(\epsilon = 0.37\), there are two \((f,g)\)-fixed points, \((0.335, 0.3202)\) and \((0.2266, 0.1795)\) – the decoding process gets stuck at \((0.335, 0.3202)\). When \(\epsilon = 0.35 = \epsilon^*_G\), there is exactly one \((f,g)\)-fixed point at \((0.27, 0.237)\), and the dashed and dotted lines osculate.

V. AN LDPCL CONSTRUCTION AND ACHIEVING CAPACITY

In this section, we present an LDPCL ensemble construction. First we assume that the local ensemble (i.e., \(\lambda_L, \rho_L\)) is given (Sub-section V-A), and then we optimize the local ensemble
Fig. 6. Illustration of the density-evolution equations in (9a)-(9c) for the LDPCL ensemble in Example 4 which induce a global decoding threshold of $\epsilon^* = 0.35$. The evolved channel erasure probabilities, from left to right, are $\epsilon = 0.33, 0.35, 0.37$.

according to the proposed construction (Sub-sections V-B and V-C). Finally, we use capacity-achieving sequences of LDPC ensembles to construct a LDPCL capacity-achieving sequence that in addition can be sub-block decoded up to the local threshold permitted by the local degree distributions. In general, inputs for the construction are the desired local and global decoding thresholds, $\epsilon_L$ and $\epsilon_G$, respectively, and the outputs are degree-distributions $(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0)$ such that

$$
\epsilon_L^* (\lambda_L, \rho_L) = \epsilon_L
$$

$$
\epsilon_G^* (\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) = \epsilon.
$$

Note that setting $P_0 = 0$, and picking any two LDPC ensembles $(\lambda_L, \rho_L)$ and $(\lambda_J, \rho_J)$ that induce thresholds $\epsilon^* (\lambda_L, \rho_L) = \epsilon_L$ and $\epsilon^* (\lambda_J, \rho_J) = \epsilon_G$ would suffice, but this choice yields poor rates (intuitively, with that choice the local and joint codes do not "cooperate"). Another solution is not using a joint ensemble at all, i.e., choosing $(\lambda_L, \rho_L)$ such that $\epsilon^* (\lambda_L, \rho_L) = \epsilon_G > \epsilon_L$, and setting $P_0 = 1$. However, this solution is an undesired overkill since it would miss the opportunity to have a low-complexity local decoder for the majority of decoding instances where the erasure probabilities are below $\epsilon_L$. 
A. The Construction

In this sub-section, we assume that the local ensemble \((\lambda_L, \rho_L)\) and the global decoding threshold \(\epsilon_G\) are given, and we give a simple way to determine the joint ensemble \((\lambda_J, \rho_J, P_0)\)

such that \(\epsilon^*_G(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) = \epsilon_G\).

**Definition 4:** Let \((\lambda_L, \rho_L)\) be local degree-distribution polynomials, and let \(\epsilon^*_L(\lambda_L, \rho_L) = \epsilon_L\) be their BP decoding threshold. For \(\epsilon \in (\epsilon_L, 1)\), let

1) \(h_\epsilon(x) = \epsilon \lambda_L(1 - \rho_L(1 - x)) - x, \quad x \in [0, 1]\)

2) \(x_s(\epsilon) = \max\{x \in [0, 1] : h_\epsilon(x) \geq 0\}\)

3) \(a_s(\epsilon) = \Lambda_L(1 - \rho_L(1 - x_s(\epsilon)))\)

For every \(x \in [0, 1]\), \(h_\epsilon(x)\) is the erasure-probability change in one iteration of the BP algorithm on the local graph, if the current erasure probability is \(x\). By definition, since \(\epsilon > \epsilon_L\), \(h_\epsilon(x) > 0\) for some \(x \in [0, 1]\). In addition, for every \(x > \epsilon\), \(h_\epsilon(x) < 0\), so \(x_s(\epsilon)\) is well defined. Operationally, \(x_s(\epsilon)\) is the local-edge erasure probability when the local decoder gets stuck. Definitions similar to items 1) and 2) have appeared in [9]; we add \(a_s(\epsilon)\) as the variable-node erasure probability corresponding to \(x_s(\epsilon)\).

**Theorem 4:** let \((\lambda_L, \rho_L)\) be a local ensemble inducing a local threshold \(\epsilon_L\), and let \(\epsilon_G \in (\epsilon_L, 1)\). If \(P_0 = \frac{\epsilon_L}{\epsilon_G}\), and \((\lambda_J, \rho_J)\) is an ensemble having a decoding threshold \(\epsilon_J = \epsilon_G \cdot a_s(\epsilon_G)\), then \(\epsilon^*_G(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) = \epsilon_G\).

**Proof:** Let \(\bar{\epsilon} < \epsilon_G\). In view of (14a), for every \(x \in (0, 1]\),

\[
f(\bar{\epsilon}, x, 0) = \bar{\epsilon} \lambda_L(1 - \rho_L(1 - x))P_0 \\
< P_0 \epsilon_G \lambda_L(1 - \rho_L(1 - x)) \\
= \epsilon_L \lambda_L(1 - \rho_L(1 - x)) \\
\leq x.
\]

Furthermore, Definition [4] implies that for every \((x, y) \in (x_s(\bar{\epsilon}), 1) \times [0, 1]\),

\[
f(\bar{\epsilon}, x, y) = \bar{\epsilon} \lambda_L(1 - \rho_L(1 - x))\Lambda_J(1 - \rho_J(1 - y)) \\
< x \Lambda_J(1 - \rho_J(1 - y)) \\
\leq x,
\]
and from Lemma 1 if \((x, y) \in (0, x_s(\bar{\epsilon})) \times (0, 1),\)
\[
g(\bar{\epsilon}, x, y) \leq g(\bar{\epsilon}, x_s(\bar{\epsilon}), y) \\
= \bar{\epsilon} \lambda_J (1 - \rho_J (1 - y)) \lambda_L (1 - \rho_L (1 - x_s(\bar{\epsilon}))) \\
= \bar{\epsilon} \lambda_J (1 - \rho_J (1 - y)) a_s(\bar{\epsilon}). \tag{31}
\]
Since \(\bar{\epsilon} < \epsilon_G,\) then \(\epsilon_J = \epsilon_G \cdot a_s(\epsilon_G) > \bar{\epsilon} \cdot a_s(\bar{\epsilon}),\) where \(\epsilon_J\) is the BP decoding threshold of \((\lambda_J, \rho_J);\) thus (31) yields
\[
g(\bar{\epsilon}, x, y) < \epsilon_J \lambda_J (1 - \rho_J (1 - y)) \leq y, \quad \forall (x, y) \in (0, x_s(\bar{\epsilon})) \times (0, 1]. \tag{32}
\]
Combining (29), (30), and (32) implies that (16) has no solution in \((0, 1] \times [0, 1).\) This in turn implies, due to Theorem 2, that \(\epsilon^*_G \geq \bar{\epsilon}.\) Since this is true for any \(\bar{\epsilon} < \epsilon_G,\) we conclude that
\[
\epsilon^*_G \geq \epsilon_G. \tag{33}
\]
Finally, from Theorem 3 and (8), since \(P_0 = \frac{\epsilon_L}{\epsilon_G} > 0,\) we have
\[
\epsilon^*_G \leq \frac{1}{P_0} \cdot \inf_{(0,1]} \frac{x}{\lambda_L (1 - \rho_L (1 - x))} = \frac{\epsilon_L}{P_0} = \epsilon_G. \tag{34}
\]

**B. LP Optimized Degree-Distributions**

From the capacity bound, the design rate of an ensemble constructed by Theorem 4 is upper bounded by
\[
R(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) < 1 - \frac{\int_0^1 \rho_L}{\int_0^1 \lambda_L} - \epsilon_J \cdot (1 - P_0). \tag{35}
\]
In view of item 3) in Definition 4, for a fixed \(\epsilon_G \in (0, 1),\) \(\epsilon_J = \epsilon_G \cdot a_s(\epsilon_G)\) is monotonically increasing w.r.t. \(x_s(\epsilon_G),\) hence the design rate could increase if \(x_s(\epsilon_G)\) decreases. On the other hand, lowering \(x_s(\epsilon_G)\) adds constraints on the local ensemble, and it may degrade the design rate (i.e., if \(\int_0^1 \frac{\rho_L}{\int_0^1 \lambda_L} \) increases).

In this sub-section we use linear programming to find a local ensemble that maximizes the bound on the design rate of the construction proposed in Theorem 4. In other words, we assume that the local and global thresholds \((\epsilon_L, \epsilon_G)\) are given and we ask: under a constraint on the maximum local degrees \(l_{max}, r_{max},\) what are the local degree-distributions that bring the bound in (35) to maximum. This maximum-degree constraint is motivated by the low-complexity requirement of the local code. Note that the joint ensemble is not limited (as long as its decoding
threshold satisfies the hypothesis of Theorem 4, and one can think of it as an ensemble that almost achieves capacity.

As usual, we iterate between optimizing the variables degree-distribution, \((\lambda_{L,1}, \ldots, \lambda_{L,l_{\text{max}}})\), and checks degree-distribution, \((\rho_{L,1}, \ldots, \rho_{L,r_{\text{max}}})\). Let \(x_s \in (0, 1)\), fix \(\rho_L(x) = x^{r_{\text{max}}-1}\), and solve

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=2}^{l_{\text{max}}} \frac{\lambda_{L,i}}{i} \\
\text{subject to} & \quad \lambda_{L,i} \geq 0, \quad i \in \{2, \ldots, l_{\text{max}}\}, \\
& \quad \sum_{i=2}^{l_{\text{max}}} \lambda_{L,i} = 1, \\
& \quad \epsilon_L \lambda_L(1 - \rho_L(1 - x)) - x \leq 0, \quad x \in (0, 1], \\
& \quad \epsilon L \lambda_L(1 - \rho_L(1 - x)) - x \leq 0, \quad x \in (x_s, 1]. 
\end{align*}
\]

(36)

With the resulting \(\lambda_L\) in hand, solve

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=2}^{r_{\text{max}}} \frac{\rho_{L,i}}{i} \\
\text{subject to} & \quad \rho_{L,i} \geq 0, \quad i \in \{2, \ldots, r_{\text{max}}\}, \\
& \quad \sum_{i=2}^{r_{\text{max}}} \rho_{L,i} = 1, \\
& \quad 1 - z - \rho_L(1 - \epsilon_L \lambda_L(z)) \leq 0, \quad z \in (0, 1], \\
& \quad 1 - z - \rho_L(1 - \epsilon_L \lambda_L(z)), \quad z \in \left(\lambda_L^{-1}(\frac{x_s}{2}), 1]\right). 
\end{align*}
\]

(37)

By iterating between (36) and (37) for different values of \(x_s \in (0, 1)\), the design rate of the ensemble constructed in Theorem 4 is maximized.

C. Low Degree Local Ensembles

In most cases, it is hard to produce an analytical expression for \(x_s(\epsilon)\), but if we limit the local degrees of the ensemble to be small, then a closed-form expression could be derived for \(x_s(\epsilon), a_s(\epsilon)\), and \(\epsilon_J\). In particular, we consider local ensembles taking the form:

\[
\lambda_L(x) = x, \quad \rho_L(x) = \rho_2x + \rho_3x^2 + \rho_4x^3, \quad \rho_i \geq 0, \quad \rho_2 + \rho_3 + \rho_4 = 1. \tag{38}
\]

This restriction to very low local degrees yields a very low-complexity local BP decoding algorithm.

Lemma 6: For the family of ensembles given in (38),

\[
\epsilon_L = \frac{1}{1 + \rho_3 + 2\rho_4}. \tag{39}
\]
In addition, for every $\epsilon \in (\epsilon_L, 1)$, if $\rho_4 = 0$, then

$$x_s(\epsilon) = 1 - \frac{1}{\rho_3} \left( \frac{1}{\epsilon} - 1 \right),$$  
(40)

and if $\rho_4 > 0$, then

$$x_s(\epsilon) = \frac{\rho_3 + 3 \rho_4 - \sqrt{(\rho_3 + \rho_4)^2 + 4 \rho_4 (\frac{1}{\epsilon} - 1)}}{2 \rho_4}. \tag{41}$$

Furthermore,

$$a_s(\epsilon) = \left( \frac{x_s(\epsilon)}{\epsilon} \right)^2. \tag{42}$$

**Proof:** See Appendix E \hfill \blacksquare

**Remark 5:** To emphasize the fact the $x_s(\epsilon)$ depends on $\rho_3, \rho_4$, we use the notation $x_s(\rho_3, \rho_4, \epsilon)$ for the rest of this sub-section.

In view of Lemma 6 in the special case where the the local ensemble is in the form of (38), the procedure of finding the optimal $x_s(\rho_3, \rho_4, \epsilon)$ described in Section V-B boils down to solving a closed-form optimization problem. Let $0 < \epsilon_L < \epsilon_G < 1$. If $(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0)$ is an ensemble such that $(\lambda_L, \rho_L)$ is in the form of (38), and $(\lambda_J, \rho_J, P_0)$ satisfies the hypotheses of Theorem 4 then from (35) and (42)

$$R < \frac{1}{3} \rho_3 + \frac{1}{2} \rho_4 - \epsilon_G \cdot a_s(\epsilon_G) \cdot \left( 1 - \frac{\epsilon_L}{\epsilon_G} \right)$$

$$= \frac{1}{3} \rho_3 + \frac{1}{2} \rho_4 - \frac{x^2_s(\rho_3, \rho_4, \epsilon_G)}{\epsilon_G} \cdot \left( 1 - \frac{\epsilon_L}{\epsilon_G} \right)$$

$$\triangleq \hat{R}(\rho_3, \rho_4), \tag{43}$$

Moreover, in order to induce a local decoding threshold at least $\epsilon_L$, Lemma 6 implies that $\rho_3 + 2 \rho_4 \leq \frac{1}{\epsilon_L} - 1$; hence, (43) yields the following two-variables optimization problem

$$\begin{align*}
\text{maximize} & \quad \hat{R}(\rho_3, \rho_4) \\
\text{subject to} & \quad \rho_3, \rho_4 \geq 0 \\
& \quad \rho_3 + \rho_4 \leq 1 \\
& \quad \rho_3 + 2 \rho_4 \leq \frac{1}{\epsilon_L} - 1, \tag{44}
\end{align*}$$

that can be solved with some standard optimization techniques.
D. Achieving Capacity

In this sub-section, we define the LDPCL notion of capacity-achieving sequences, and prove that such sequences exist. During the derivation, we will refer to \( \delta(\lambda, \rho) \) as the additive gap to capacity of the \( LDPC(\lambda, \rho) \) ensemble, i.e., \( \delta(\lambda, \rho) = 1 - \epsilon^*(\lambda, \rho) - R(\lambda, \rho) \). Similarly, we define \( \delta(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) = 1 - \epsilon_G^*(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) - R(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) \) as the global additive gap to capacity.

**Definition 5:** Let \( 0 < \epsilon_L < \epsilon_G < 1 \). A sequence \( \{\lambda^{(k)}_L, \rho^{(k)}_L, \lambda^{(k)}_J, \rho^{(k)}_J, P_0^{(k)}\}_{k \geq 1} \) is said to achieve capacity on a \( BEC(\epsilon_G) \), with local decoding capability \( \epsilon_L \) if:

1. \( \lim_{k \to \infty} \epsilon^*_L \left( \lambda^{(k)}_L, \rho^{(k)}_L \right) = \epsilon_L \)
2. \( \lim_{k \to \infty} \epsilon^*_G \left( \lambda^{(k)}_L, \rho^{(k)}_L, \lambda^{(k)}_J, \rho^{(k)}_J, P_0^{(k)} \right) = \epsilon_G \)
3. \( \lim_{k \to \infty} R \left( \lambda^{(k)}_L, \rho^{(k)}_L, \lambda^{(k)}_J, \rho^{(k)}_J, P_0^{(k)} \right) = 1 - \epsilon_G \)

Note that Items 2) and 3) imply that \( \lim_{k \to \infty} \delta \left( \lambda^{(k)}_L, \rho^{(k)}_L, \lambda^{(k)}_J, \rho^{(k)}_J, P_0^{(k)} \right) = 0 \).

**Lemma 7:** Let \( (\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) \) be degree-distribution polynomials constructed according to Theorem 4 and let \( \delta_L \equiv \delta(\lambda_L, \rho_L) \) and \( \delta_J \equiv \delta(\lambda_J, \rho_J) \). Then,

\[
\delta(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) \leq \delta_L + \delta_J \cdot (1 - P_0).
\]  

**Proof:** Let \( \epsilon_G = \epsilon_G^*(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) \) be the BP global decoding threshold of the \( LDPCCL(M, n, \lambda_L, \rho_L, \lambda_J, \rho_J, P_0) \) ensemble as \( n \to \infty \). In view of Theorem 4, \( P_0 = \frac{\epsilon_f}{\epsilon_G} \), and

\[
\epsilon_J \equiv \epsilon^*(\lambda_J, \rho_J) = \epsilon_G \cdot a_s(\epsilon_G) \leq \epsilon_G.
\]  

In addition, by definition we have,

\[
\begin{align*}
\frac{\int_0^1 \rho_L(x)dx}{\int_0^1 \lambda_L(x)dx} &= \epsilon_L + \delta_L \\
\frac{\int_0^1 \rho_J(x)dx}{\int_0^1 \lambda_J(x)dx} &= \epsilon_J + \delta_J,
\end{align*}
\]  

hence (7), (46) and (47) imply,

\[
\begin{align*}
\delta(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) &= 1 - R(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) - \epsilon_G^*(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0) \\
&= \epsilon_L + \delta_L + (\epsilon_J + \delta_J) \left( 1 - \frac{\epsilon_L}{\epsilon_G} \right) - \epsilon_G \\
&= \frac{1}{\epsilon_G} \left( \frac{\epsilon_J - \epsilon_G}{\epsilon_G - \epsilon_L} \right) + \delta_L + \delta_J \left( 1 - \frac{\epsilon_L}{\epsilon_G} \right) \\
&\leq \delta_L + \delta_J \left( 1 - \frac{\epsilon_L}{\epsilon_G} \right)
\end{align*}
\]  

(48)
\[
\delta_L + \delta_J (1 - P_0).
\] (49)

At this point, it is clear how to construct a capacity-achieving sequence of LDPCL ensembles on a BEC(\(\epsilon_G\)), with a local decoding capability \(\epsilon_L\). Choose any two sequences of (ordinary) LDPC ensembles \(\{\lambda_L^{(k)}, \rho_L^{(k)}\}_{k \geq 1}\) and \(\{\lambda_J^{(k)}, \rho_J^{(k)}\}_{k \geq 1}\) that achieve capacity on the BEC(\(\epsilon_L\)) and BEC(\(\epsilon_G\)), respectively, and set \(P_0^{(k)} = \left(1 - \frac{\epsilon_L}{\epsilon_G}\right)\), for all \(k \geq 1\). Item 1) in Definition 5 clearly holds for this sequence, and in view of Theorem 4 Item 2) in Definition 5 holds as well. Finally, Lemma 7 implies that
\[
\lim_{k \to \infty} \delta \left(\lambda_L^{(k)}, \rho_L^{(k)}, \lambda_J^{(k)}, \rho_J^{(k)}, P_0\right) \leq \lim_{k \to \infty} \delta \left(\lambda_L^{(k)}, \rho_L^{(k)}\right) + \lim_{k \to \infty} \delta \left(\lambda_J^{(k)}, \rho_J^{(k)}\right) \left(1 - \frac{\epsilon_L}{\epsilon}\right) = 0.
\]

**Example 5:** We construct an LDPCL capacity-achieving sequence with local and global threshold \(\epsilon_L = 0.05\) and \(\epsilon_G = 0.2\), respectively. We set \(P_0 = \frac{\epsilon_L}{\epsilon_G} = 0.25\) and we use the Tornado capacity-achieving sequence \([11]\),

\[
\lambda^{(D_L)}_L(x) = \frac{1}{H(D_L)} \sum_{i=1}^{D_L} \frac{x^i}{i}, \quad \lambda^{(D_J)}_J(x) = \frac{1}{H(D_J)} \sum_{i=1}^{D_J} \frac{x^i}{i},
\]

\[
\rho^{(D_L)}_L(x) = e^{-\alpha_L} \sum_{i=0}^{\infty} \frac{\lambda^{(D_L)}_L(x)^i}{i!}, \quad \rho^{(D_J)}_J(x) = e^{-\alpha_J} \sum_{i=0}^{\infty} \frac{\lambda^{(D_J)}_J(x)^i}{i!},
\] (50)

where \(H(\cdot)\) is the harmonic sum, \(\alpha_L = \frac{H(D_L)}{\epsilon_L}\), and \(\lambda^{(D_J)}_J(x), \rho^{(D_J)}_J(x)\) are defined similarly (the check degree-distribution series are truncated to get degree-distribution polynomials with finite degrees). \(D_L\) (resp. \(D_J\)) controls the local (resp. joint) gap to capacity \(\delta_L\) (resp. \(\delta_J\)); the bigger it is, the smaller the gap is. Table I exemplifies how the LDPCL sequence \(\{\lambda^{(D_L)}, \rho^{(D_L)}_L, \lambda^{(D_J)}, \rho^{(D_J)}_J, P_0\}\) approaches capacity as \(D_L \to \infty, D_J \to \infty\): Theorem 4 implies that for every value of \(D_L\) and \(D_J\), the global decoding threshold is \(\epsilon^*_G \geq 0.2\); the local additive gap to capacity \(\delta_L\) and joint additive gap to capacity \(\delta_J\) both vanish as \(D_L \to \infty\) and \(D_J \to \infty\), which in view of (45), implies that the global additive gap to capacity \(\delta\) vanishes as well.

Table I shows the advantage of the multi-block scheme: one can achieve the local threshold with significantly simpler ensembles (use only few terms in (50)). Such simpler codes allow to approach the theoretical threshold with shorter blocks and with lower complexity.

**VI. REDUCING THE NUMBER OF JOINT ITERATIONS**

It has not been emphasized earlier in the paper, but in practical settings the local and joint decoding iterations are very different in terms of cost. Joint iterations access a much larger (factor
In distributed-storage applications the sub-blocks may even reside in remote sites, increasing the communication cost even further. Therefore, we would like to reduce the number of joint iterations (call it $N_{JI}$) performed by the decoder (i.e., rounds of variable-to-joint-check messages and joint-check-to-variable messages). Ideally, the decoder successfully decodes the desired sub-block (of length $n$) on the local graph (see Section II-B), does not enter the global mode, and no joint iterations are needed ($N_{JI} = 0$); this happens when the fraction of erased bits is equal or less than the local threshold, i.e., $\epsilon \leq \epsilon_L$. However, if $\epsilon > \epsilon_L$, then the decoder fails to decode in the local mode and it enters the global mode where at least one joint iteration is necessary ($N_{JI} \geq 1$).

In this section, we suggest a scheduling scheme for updating the joint side of the two-sided Tanner graph during the global mode decoding, and we prove that it is optimal in the sense of minimizing $N_{JI}$. In addition, we study how the parameters of the local and joint ensembles affect $N_{JI}$.

### A. An $N_{JI}$-optimal scheduling scheme

Recall the density-evolution equations for the BP decoding algorithm over the BEC with erasure probability $\epsilon$, of the $LDPCL(M, n, \Lambda_L, \Lambda_J, \Omega_L, \Omega_J)$ ensemble when $n \to \infty$ :

\[ x_l = f(\epsilon, x_{l-1}, y_{l-1}), \quad l \geq 0, \tag{51a} \]
\[ y_l = g(\epsilon, x_{l-1}, y_{l-1}), \quad l \geq 0, \tag{51b} \]
\[ x_{-1} = y_{-1} = 1, \tag{51c} \]

\[ M \) data unit, which may involve a high cost of transferring the bits to the check-node logic. 

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
$D_L$ & $\delta_L$ & $D_J$ & $\delta_J$ & Rate \\
\hline
1 & 0.05 & 1 & 0.2 & 0.6 \\
1 & 0.05 & 2 & 0.1 & 0.67 \\
1 & 0.05 & 10 & 0.02 & 0.735 \\
1 & 0.05 & 100 & 0.002 & 0.745 \\
2 & 0.025 & 100 & 0.002 & 0.775 \\
5 & 0.01 & 100 & 0.002 & 0.79 \\
$\infty$ & 0 & $\infty$ & 0 & 0.8 \\
\hline
\end{tabular}
\caption{Local gap to capacity $\delta_L$, joint gap to capacity $\delta_J$, and rate of the LDPCL capacity-achieving sequence given by \eqref{eq:lin_opt} with $\epsilon_L = 0.05$, $\epsilon_G = 0.2$, and $P_0 = 0.25$.}
\end{table}
where \( f \) and \( g \) are given in (14a) and (14b), respectively; note that (51a) and (51b) express a local and a joint iteration, respectively. A scheduling scheme prescribes decoder access to the joint check nodes in only part of the iterations, and thus replaces (51b) with

\[
y_l = \begin{cases} 
g(\epsilon, x_{l-1}, y_{l-1}) & l \in A \\
y_{l-1} & l \notin A \end{cases}
\]

for some \( A \subseteq \mathbb{N} \) representing the iteration numbers where joint checks are accessed; in this case we have

\[
N_{JI} = |A|.
\]

Note that when the check-node update is skipped in the joint side there is no need to access the variable nodes outside the sub-block. Since Lemma 3 still holds when (51b) is replaced with (52), the limits \( \lim_{l \to \infty} x_l \) and \( \lim_{l \to \infty} y_l \) exist for every scheduling scheme.

Given local and joint degree-distributions, a scheduling scheme is called valid if for every \( \epsilon < \epsilon_G = \epsilon^*_{G}(\Lambda_L, \Lambda_J, \Omega_L, \Omega_J) \), \( \lim_{l \to \infty} x_l(\epsilon) = 0 \) (successful decoding). Our goal is to find an optimal scheduling scheme: a valid scheduling scheme that minimizes \( N_{JI} \). For example, if \( A = \emptyset \) (no joint updates), then \( N_{JI} = 0 \) but \( \lim_{l \to \infty} x_l(\epsilon) > 0 \) if \( \epsilon \in (\epsilon_L, \epsilon_G) \); thus, the scheduling scheme is not valid. If, on the other hand, \( A = \mathbb{N} \) (joint checks are accessed in every iteration), then the scheduling scheme is valid, but \( N_{JI} \) equals the total number of iterations, which is the worst case. We do not require the scheduling scheme to be pre-determined, and it can use “on-line” information about the decoding process, for example, the current fraction of erasure messages or the change in this fraction between two consecutive iterations. In what follows, we present an optimal scheduling scheme that minimizes \( N_{JI} \) while assuring that \( \lim_{l \to \infty} x_l = 0 \).

**Definition 6:** Let \( (\Lambda_J, \rho_J) \) be joint degree-distribution polynomials, let \( \epsilon \in (0,1) \) be the erasure probability of a \( BEC \), and let \( y \in [0,1] \) be an instantaneous erasure probability from the joint perspective. We define the effective erasure probability from the local perspective as

\[
\epsilon_{loc}(y) = \epsilon \cdot \Lambda_J(1 - \rho_J(1 - y)).
\]

In view of (14a) and (53), we have

\[
x_l = f(\epsilon, x_{l-1}, y_{l-1}) = \epsilon_{loc}(y_{l-1})\lambda_L(1-\rho_L(1 - x_{l-1})).
\]

\( \epsilon_{loc}(y_{l-1}) \) takes the role of \( \epsilon \) when the local code is viewed as a standard LDPC code, hence the term “effective erasure probability from the local perspective”.

\[
\epsilon_{loc}(y_{l-1}) = \epsilon \cdot \Lambda_J(1 - \rho_J(1 - y_{l-1})).
\]
Our scheduling scheme is parametrized by \( \eta > 0 \), and is given by

\[
x_l = f(\epsilon, x_{l-1}, y_{l-1}) \\
y_l = \begin{cases} 
g(\epsilon, x_{l-1}, y_{l-1}) & |x_{l-2} - x_{l-1}| \leq \eta \text{ and } \epsilon_{\text{loc}}(y_{l-1}) \geq \epsilon_L \\
y_{l-1} & \text{else}
\end{cases}
\]  \tag{55}

**Lemma 8:** For every \( \eta > 0 \), the scheduling scheme described in (55) is valid.

**Proof:** See Appendix F. \hfill \blacksquare

Note that if \( \eta = 0 \), the scheduling scheme described in (55) is not valid. However, since local iterations have zero cost in our model, we can assume that we can apply arbitrarily many local iterations to get arbitrarily close to \( \eta = 0 \). Simulations show that \( \eta = 10^{-4} \) suffices for achieving minimal \( N_{JI} \). Thus, for the following analysis we will assume that \( \eta = 0 \), and that the scheduling scheme is still valid. In this scheduling scheme, the decoder tries to decode the sub-block on the local graph until it gets “stuck”, which refers to not being able to reduce the erasure probability while still strictly greater than zero. This happens first when \( x_{l_1} = x_s(\epsilon) \), for some iteration \( l_1 \), where \( x_s(\epsilon) \) is given in Definition \( \[ \] \). So, in the first joint update we have

\[
x_{l_1} = x_s(\epsilon), \quad y_{l_1} = 1
\]

\[x_{l_1+1} = x_s(\epsilon), \quad y_{l_1+1} = g(\epsilon, x_s(\epsilon), 1).
\]

In view of (54), the local graph now "sees" \( \epsilon_2 \triangleq \epsilon_{\text{loc}}(y_{l_1+1}) < \epsilon \) as an effective erasure probability, and it will proceed the decoding algorithm locally. It may get "stuck" again and another joint update will be invoked; this procedure continues until \( \epsilon_{\text{loc}}(y_{l_p+1}) < \epsilon_L \) in the \( p \)’th (and last) update, which enables successful local decoding (i.e., \( N_{JI} = p \)). In general, let \( \{l_k\}_{k=1}^{N_{JI}} \) be the joint update iterations of the scheduling scheme described above and let \( \epsilon_k \) be the effective erasure probability from the local perspective between joint updates \( i-1 \) and \( i \).

Then,

\[
y_{l_i} = 1, \quad \epsilon_1 = \epsilon, \quad x_{l_i} = x_s(\epsilon), \tag{56a}
\]

\[
y_{l_k} = g(\epsilon, x_{l_{k-1}}, y_{l_{k-1}}), \quad 2 \leq k \leq N_{JI}, \tag{56b}
\]

\[
\epsilon_k = \epsilon_{\text{loc}}(y_{l_k}), \quad 2 \leq k \leq N_{JI}, \tag{56c}
\]

\[
x_{l_k} = x_s(\epsilon_k), \quad 2 \leq k \leq N_{JI}, \tag{56d}
\]

where

\[
\epsilon = \epsilon_1 > \epsilon_2 > \ldots > \epsilon_{N_{JI}-1} \geq \epsilon_L > \epsilon_{N_{JI}}, \tag{57}
\]
Note that \((56b)\) holds for every scheduling scheme, but \((56d)\) holds only for the suggested scheduling scheme; in view of Definition \([4]\) Item 2 for any other scheme
\[x_{ik} \geq x_s(\epsilon_k), \quad 1 \leq k \leq N_{JI}.\] (58)

Lemma 9: The scheduling scheme in \((56a)-(56d)\) is optimal.

Proof: See Appendix \([G]\) \[\blacksquare\]

We assume from now on that the decoder applies the optimal scheduling scheme in \((56a)-(56d)\).

B. The Rate vs. \(N_{JI}\) Trade-Off

As shown in Section \([V-D]\) for every \(0 < \epsilon_L < \epsilon < 1\), there exist capacity-achieving LDPCL sequences for the \(BEC(\epsilon)\) with a local decoding threshold of \(\epsilon_L\). Moreover, Theorem \([4]\) and Lemma \([7]\) provide a construction of such a sequence. However, the closer an LDPCL ensemble is to capacity, the higher the \(N_{JI}\) is; therefore, to decrease \(N_{JI}\) we have to pay with rate, and there are several ways to do so. In this section we study how the parameters of the local and joint degree-distributions, \(\lambda_L, \rho_L, \lambda_J, \rho_J, P_0\), affect \(N_{JI}\). In particular, we focus on how the local and joint additive gaps to capacity \(\delta_L\) and \(\delta_J\), receptively, affect \(N_{JI}\).

It is well known that if \(\{\lambda^{(k)}(\epsilon_L), \rho^{(k)}(\epsilon_L)\}_{k=1}^{\infty}\) is a (ordinary) capacity-achieving sequence for the \(BEC(\epsilon)\), then
\[\lim_{k \to \infty} \epsilon \lambda^{(k)}(1 - \rho^{(k)}(1 - x)) = x, \quad x \in [0, \epsilon].\] (59)

This leads to the following lemma.

Lemma 10: Let \(0 < \epsilon_L < \epsilon < 1\), and let \(\{\lambda^{(k)}(\epsilon_L), \rho^{(k)}(\epsilon_L)\}_{k=1}^{\infty}\) be a capacity-achieving sequence for the \(BEC(\epsilon_L)\). Then,
\[x_s(\epsilon) \triangleq \lim_{k \to \infty} x^{(k)}_s(\epsilon) = \epsilon,\]
where \(x^{(k)}_s(\epsilon)\) corresponds to Definition \([4]\) with \(\{\lambda^{(k)}(\epsilon_L), \rho^{(k)}(\epsilon_L)\}\).

Proof: See Appendix \([H]\) \[\blacksquare\]

Lemma \([10]\) asserts that if the local degree-distribution polynomials imply a local threshold \(\epsilon_L\) and a design rate that is very close to capacity \((1 - \epsilon_L)\), and the channel erasure probability is \(\epsilon > \epsilon_L\), then the BP decoding algorithm on the local graph gets “stuck” immediately after solving only a small fraction of the erasures. This leads, in view of \((56b)\), to a small change in the erasure-message probability on the joint update, which in turn yields a minor progress
on the local side. Therefore, choosing close to capacity local degree-distribution polynomials implies high $N_{JI}$.

Another consequence of (59) is that the change in the erasure-message probability in one iteration of the BP decoding algorithm is small. Thus, close to capacity joint degree-distribution polynomials yield high $N_{JI}$, regardless of the local degree-distribution polynomials.

Example 6: Let $\epsilon_L = 0.05$ and $\epsilon_G = 0.2$. We use the capacity-achieving LDPCL sequence for the $BEC(\epsilon_G)$ with a local decoding threshold of $\epsilon_L$ given in Example 5.

A computer program simulated (56a)-(56d) with (50) for different values of $D_L$ and $D_J$, and the results are presented in Figure 7. The plot exemplifies the trade-off between rate and $N_{JI}$: when the ensemble is close to capacity with $\delta_L = 10^{-2}, \delta_J = 2.5 \cdot 10^{-4}$ ($R = 0.79$), we get $N_{JI} = 570$, and to reduce $N_{JI}$ we have to pay with rate. However, there are several ways to do so. For example, changing in the above the local gap to $\delta_L = 5 \cdot 10^{-2}$ while the joint gap stays $\delta_J = 2.5 \cdot 10^{-4}$ yields $R = 0.75$ and $N_{JI} = 26$, and changing the local and joint gap to $\delta_L = 2.5 \cdot 10^{-2}$ and $\delta_J = 4 \cdot 10^{-2}$, respectively, yields the same $R = 0.75$ but a smaller $N_{JI} = 11$.

![Fig. 7. Plot of $N_{JI}$ (○ marks) and the design rate (+ marks) as a function of the joint additive gap to capacity $\delta_J$ for different value of the local additive gap to capacity $\delta_L$.](image-url)
VII. FINITE-LENGTH ANALYSIS FOR ML DECODING

In this section, we extend to LDPCL codes the finite block-length analysis of [3, Lemma B.2]. We derive upper bounds on the expected block erasure probability under ML decoding of regular LDPCL ensembles (see Section III-A). As shown in [3], the union bound is fairly tight for not too small block lengths, and the performance loss of iterative decoding compared to ML decoding is not too high. This motivates a bounding approach that simplifies the expressions by employing the union bound and the ML decoding analysis.

For a two-sided Tanner graph \( G \) picked from the \((M,n,l,L,r,J)\)-regular ensemble, let \( P_{B}^{\text{ML}}(G,\epsilon) \) designate the expected block decoding failure probability of the code \( G \) over the \( \text{BEC}(\epsilon) \) when decoded by the ML decoder. In what follows, for a power series \( f(x) = \sum_{i} f_{i} x^{i} \), let \( \text{coef}(f(x),x^{i}) \) designate the \( i \)-th coefficient of \( f(x) \), i.e., \( \text{coef}(f(x),x^{i}) = f_{i} \).

**Definition 7:** For every \( l,r,n,w \in \mathbb{N} \) such that \( n,M \in \mathbb{N} \), \( l,r,n \geq 1 \) and \( w \leq n \), let

\[
A(l,r,n,w) = \text{coef}\left(\left(\frac{(1+x)^{l}+(1-x)^{r}}{2^{n}}\right)^{n/M},x^{w}\right). \tag{60}
\]

**Theorem 5 (Union Bound - ML Decoding of regular LDPCL Codes):**

1) For \( \epsilon \in (0,1) \), let \( \bar{\epsilon} = 1 - \epsilon \).
2) For \( n \in \mathbb{N^{+}} \), let \([n] = \{1,2,\ldots,n\}\), and \([n]_{0} = \{0\} \cup [n]\).
3) For \( m \in \mathbb{N} \), let \( e_{1}^{m} = (e_{1},e_{2},\ldots,e_{m}) \), where for every \( i \in \{1,2,\ldots,m\}, e_{i} \in \mathbb{N^{+}} \), and let \([e_{1}^{m}]_{0} = [e_{1}]_{0} \times [e_{2}]_{0} \times \cdots \times [e_{m}]_{0} \).

Then,

\[
E\left[\mathbb{P}_{B}^{\text{ML}}(G,\epsilon)\right] \leq \sum_{m=1}^{M} \binom{M}{m} \bar{\epsilon}^{(M-m)n} \sum_{e_{i}^{m} \in [n]^{m}} \prod_{i=1}^{m} \binom{n}{e_{i}} \epsilon_{i}^{e_{i}(n-e_{i})} \cdot \min \left\{1,-1 + \sum_{w_{i}^{m} \in [e_{i}^{m}]_{0}} A(l_{i},r_{i},Mn,w_{1}+w_{2}+\cdots+w_{m}) \prod_{i=1}^{m} A(l_{L},r_{L},n,w_{i})\right\} \tag{61}
\]

where the expectation is w.r.t. the \((M,n,l,L,r,J)\)-regular ensemble.

**Proof:** Let \( E \subseteq \{1,2,\ldots,Mn\} \) be an erasure pattern specifying the indices of the erased code bits, and for every \( i \in \{1,2,\ldots,M\} \), let \( E_{i} = E \cap \{1+(i-1)n,2+(i-1)n,\ldots,n+(i-1)n\} \) be the erasure pattern in the \( i \)-th sub-block. Consider the case that there are erasures only in the
first \( m \) sub-blocks, i.e.,
\[
|\mathcal{E}_i| > 0, \quad 1 \leq i \leq m \\
|\mathcal{E}_i| = 0, \quad l < m \leq M.
\] (62)

It is well known that the ML decoder fails if and only if \( \text{rank}(H_\mathcal{E}) < |\mathcal{E}| \), where \( H \) is the parity-check matrix corresponding to a random member of the \((l_L, r_L, l_J, r_J)\)-regular LDPCL ensemble, and \( H_\mathcal{E} \) is the sub-matrix comprised from the columns of \( H \) indexed by \( \mathcal{E} \). In view of (1), since the sub-matrices \( \{H_i\}_{i=0}^{M} \) are picked independent of each other, then applying the union bound yields
\[
Pr\{\text{rank}(H_\mathcal{E}) < |\mathcal{E}|\} = Pr\left\{\exists x \in \mathbb{F}_2^{|\mathcal{E}|} \setminus \{0\}, H_\mathcal{E}x^T = 0\right\}
\leq \sum_{x \in \mathbb{F}_2^{|\mathcal{E}|} \setminus \{0\}} Pr\{H_\mathcal{E}x^T = 0\}
= -1 + \sum_{x \in \mathbb{F}_2^{|\mathcal{E}|}} \cdots \sum_{x_m \in \mathbb{F}_2^{|\mathcal{E}_m|}} Pr\{H_{0\mathcal{E}}(x_1, \ldots, x_l)^T = 0\}
\cdot \prod_{i=1}^{m} Pr\{H_{i\mathcal{E}}x_i^T = 0\}
\] (63a)

The steps until getting to (63c) are the standard ones from [3], while (63d) follows from the sub-block structure introduced in LDPCL codes. By marking \( w_i = w(x_i) \) for every \( i \in \{1, 2, \ldots, m\} \), as the Hamming weight of the sub-word \( x_i \in \mathbb{F}_2^{|\mathcal{E}_i|} \) we get (see [3 Lemma B.2.])
\[
Pr\{H_{0\mathcal{E}}(x_1, x_2, \ldots, x_l)^T = 0\} = A(l_J, r_J, M, w_1 + w_2 + \ldots + w_l).
\] (64)

Combining (63d) and (64) implies that if (62) holds, then
\[
Pr\{\text{rank}(H_\mathcal{E}) < |\mathcal{E}|\} \leq
-1 + \sum_{w_1=0}^{|\mathcal{E}_1|} \sum_{w_2=0}^{|\mathcal{E}_2|} \cdots \sum_{w_m=0}^{|\mathcal{E}_m|} A(l_J, r_J, M, w_1 + w_2 + \ldots + w_l) \prod_{i=1}^{m} A(l_L, r_L, n, w_i).
\] (65)

Since the BEC is memoryless, then
\[
Pr\{\mathcal{E}\} = \epsilon^{(M-m)n} \prod_{i=1}^{m} \epsilon^{|\mathcal{E}_i|(n-|\mathcal{E}_i|)}.
\] (66)

From symmetry, (65) holds for every error pattern \( \mathcal{E} \) that lies in exactly \( m \) sub-blocks (i.e., not only the first \( m \) sub-blocks). Finally, summing over \( m \in \{1, 2, \ldots, M\} \) and counting every error...
pattern imply

\[
E \left[ \Phi_B^{ML} (\mathcal{G}, \epsilon) \right] = \sum_{\mathcal{E}} \Pr \{ \mathcal{E} \} \Pr \{ \text{rank} (H_{\mathcal{E}}) < |\mathcal{E}| \}
\]

\[
\leq \sum_{m=1}^{M} \binom{M}{m} \epsilon^{(M-m)n} \sum_{e_i^m \in [n]^m} \prod_{i=1}^{m} \binom{n}{e_i} \epsilon_{e_i}(n-e_i)
\]

\[
\cdot \min \left\{ 1, -1 + \sum_{w_1^m \in [e_1^m]_0} A(l_J, r_J, M n + w_1 + w_2 + \ldots + w_l) \prod_{i=1}^{m} A(l_i, r_L, n, w_i) \right\}.
\]

(67)

Figure 8 plots the upper bound in (61) for the \((2, 6, 1, 6)\)-regular LDPCL ensemble, with locality \(M = 2, 3\) and sub-block length \(n = 180, 120\), respectively. Also shown is the upper bound in [3, Lemma B.2] for the standard \((3, 6)\)-regular LDPC ensemble. The ensembles in Figure 8 have the same total block length of 360 (in view of Corollary 1, the iterative decoding performance of the \((2, 6, 1, 6)\)-regular LDPCL ensemble and the \((3, 6)\)-regular LDPC ensemble coincide as \(n \to \infty\)). Figure 8 exemplifies the trade-off between the sub-block local-access capability \((M)\) and the error correcting capability over the global block; for every \(\epsilon \in [0.05, 0.5]\) the block decoding failure probability increases as \(M\) increases. While Figure 8 is just an example with not very realistic parameters, we expect the same trade-off to apply in more generality: finer sub-block access with efficient local decoding has a cost in terms of the global-decoding performance (for the same rate and global code length).

VIII. Summary

This paper lays out the theoretical foundation for multi-block LDPC codes. This coding scheme enables fast read access to small blocks, and provides high data-protection on large blocks in case of more severe error events. The decoding algorithm suggested is the belief-propagation algorithm, and by analyzing the asymptotic (as the sub-block length \(n\) tends to infinity) performance of this decoding algorithm on LDPCL codes, we provided a simple way to construct capacity-achieving sequences that enjoy a locality property (Theorem 4 and Lemma 7). We found that the fraction of jointly-unconnected variable nodes, \(P_0\), plays an important role in determining the asymptotic decoding threshold and in achieving capacity.

1Due to the combinatorial nature of the expression in (59), it is computationally difficult to evaluate the bounds for longer codes.
It is known [16] that there is an inherent trade-off between the gap to capacity and the encoding and decoding complexity of irregular LDPC codes over the BEC. Since it is of interest that the decoding algorithm will avoid joint updates as much as possible, we studied another trade-off regarding the number of joint iterations and the gap to capacity (see Example 6).

By deriving an upper bound on the block-erasure probability of finite-length LDPCL codes under ML decoding, we showed how the loss in performance is affected by the locality structure.

Future work is needed toward generalizing the results of this paper to other codes defined on graphs, such as Repeat-Accumulate (RA) [4], [8] codes or Accumulate-Repeat-Accumulate (ARA) codes [13], which achieve capacity on the BEC with bounded complexity per information bit. Future studies can also be on considering memoryless binary-input output-symmetric (MBIOS) channels other than the BEC.
APPENDIX A

PROOF OF LEMMA 2

1) Assume that \( x = 0 \). Since \( \Lambda_L(0) = 0 \), (14b) implies that \( y = g(\epsilon, 0, y) = 0 \). Moreover, if \( y = 0 \) and \( P_0 = 0 \), then (14a) yields \( x = f(\epsilon, x, 0) = 0 \). Finally, if \( y = 0 \) and \( \lambda_J(0) > 0 \), then (14b) implies that \( 0 = \Lambda_L(1 - \rho_L(1 - x)) \), hence \( \rho_L(1 - x) = 1 \) and \( x = 0 \).

2) Follows immediately from (14a), (14b) and (16).

3) We prove (17) by a mathematical induction. For \( l = 0 \), (17) holds due to Item 2 and the fact that \( x_0 = y_0 = \epsilon \). Assume correctness of (17) for some \( l \geq 0 \) and consider iteration \( l + 1 \). In view of Lemma 1, eq. (15) and the induction assumption, it follows that

\[
\begin{align*}
x_{l+1} &= f(\epsilon, x_l, y_l) \geq f(\epsilon, x, y) = x, \\
y_{l+1} &= g(\epsilon, x_l, y_l) \geq g(\epsilon, x, y) = y.
\end{align*}
\]

This prove correctness of (17) for \( l + 1 \) and by mathematical induction proves (17) for all \( l \geq 0 \).

APPENDIX B

PROOF OF LEMMA 4

Let \( I = \min \{ i : \Lambda_{L,i} > 0 \} \). Since \( \Lambda_L(0) = 0 \), then \( I \geq 1 \). The connection between \( \lambda_L(\cdot) \) and \( \Lambda_L(\cdot) \) in (4a) yields

\[
\begin{align*}
\lim_{u \to 0} \frac{\Lambda_L(u)}{\lambda_L(u)} &= \lim_{u \to 0} \frac{\Lambda_L(u)}{\Lambda'_L(u)} \cdot \Lambda'(1) \\
&= \Lambda'(1) \lim_{u \to 0} \frac{\sum_{i \geq I} \Lambda_{L,i} u^i}{u^{I-1} \sum_{i \geq I} i \Lambda_{L,i} u^{i-1}} \\
&= \Lambda'(1) \lim_{u \to 0} \frac{u^I \sum_{i \geq I} \Lambda_{L,i} u^{i-I}}{u^{I-1} \sum_{i \geq I} i \Lambda_{L,i} u^{i-I}} \\
&= \Lambda'(1) \frac{\Lambda_{L,I}}{I \Lambda_{L,I}} \lim_{u \to 0} u \\
&= 0.
\end{align*}
\]

Further, let \( u(x) = 1 - \rho_L(1 - x) \) and note that \( \lim_{x \to 0} u(x) = 0 \). Thus,

\[
\lim_{x \to 0} x \cdot \frac{\Lambda_L(1 - \rho_L(1 - x))}{\lambda_L(1 - \rho_L(1 - x))} = \lim_{x \to 0} x \cdot \frac{\Lambda_L(u(x))}{\lambda_L(u(x))} = \lim_{x \to 0} x \cdot \lim_{u \to 0} \frac{\Lambda_L(u)}{\lambda_L(u)} = 0.
\]
APPENDIX C
PROOF OF LEMMA 3

Let $\epsilon \in (0, 1)$ and let $(x, y)$ be a solution to (16) with $y > 0$. In view of (14a) and (14b), dividing the first equation of (16) with the second one yields

$$
\frac{x}{y} = \frac{\lambda_J (1 - \rho_J (1 - x)) \cdot \Lambda_J (1 - \rho_J (1 - y))}{\lambda_J (1 - \rho_J (1 - y)) \cdot \Lambda_J (1 - \rho_J (1 - x))}
$$

which after some rearrangements implies

$$
q_J(y) = q_J(x), \quad (71)
$$

where $q_J(\cdot)$ and $q_L(\cdot)$ are defined in (24). In view of (24), since $(x, y)$ is an $(f, g)$-fixed point, then

$$
q_J(y) = y \cdot \frac{\Lambda_J (1 - \rho_J (1 - y))}{\lambda_J (1 - \rho_J (1 - y))}
= g(\epsilon, x, y) \cdot \frac{\Lambda_J (1 - \rho_J (1 - y))}{\lambda_J (1 - \rho_J (1 - y))}
= \epsilon \cdot \Lambda_J (1 - \rho_J (1 - y)) \cdot \Lambda_L (1 - \rho_L (1 - x)) \leq 1,
$$

which together with Definition 3 and (71) completes the proof.

APPENDIX D
PROOF OF THEOREM 3

Let

$$
\epsilon > \inf_{y \in (0, 1]} \frac{y}{g(1, q(y), y)}, \quad (73)
$$

There exists $y_0 \in (0, 1]$ such that $q_J(y_0) \leq 1$ and

$$
y_0 = \epsilon \cdot g(1, q(y_0), y_0) = \epsilon \cdot \lambda_J (1 - \rho_J (1 - y_0)) \cdot \Lambda_L (1 - \rho_L (1 - q(y_0))). \quad (74)
$$

In view of (25),

$$
q_L(q(y_0)) = q_J(y_0), \quad (75)
$$
which combined with (14a) and (24) yields
\[
q(y_0) = \frac{\lambda_L (1 - \rho_L (1 - q(y_0)))}{\Lambda_L (1 - \rho_L (1 - q(y_0)))} \cdot y_0 \cdot \frac{\lambda_J (1 - \rho_J (1 - y_0))}{\Lambda_J (1 - \rho_J (1 - y_0))} \\
= \epsilon \cdot \lambda_L (1 - \rho_L (1 - q(y_0))) \cdot \Lambda_J (1 - \rho_J (1 - y_0)) = f(\epsilon, q(y_0), y_0). \tag{76}
\]

Thus, \((q(y_0), y_0)\) is a non zero \((f, g)\)-fixed point, which in view of Theorem 2 implies that \(\epsilon > \epsilon^*_G\). Hence,
\[
\epsilon^*_G \leq \inf_{\substack{y \in [0,1] \\ q_J(y) \leq 1}} y \cdot g(1, q(y), y). \tag{77}
\]

Next, let
\[
\epsilon < \inf_{\substack{y \in [0,1] \\ q_J(y) \leq 1}} y \cdot g(1, q(y), y) \tag{78}
\]

and let \((x, y)\) be a solution to (16). In what follows, we prove that \(y = 0\). Assume to the contrary that \(y > 0\). From Lemma 5 it follows that \(x \leq q(y)\), which in view Lemma 1 (72) and (78) implies
\[
y = g(\epsilon, x, y) \leq g(\epsilon, q(y), y) < y, \tag{79}
\]
in contradiction; thus, \(y = 0\). Next, consider two cases:

1) If \(P_0 = 0\) or \(\lambda_J(0) > 0\), then Item 1 of Lemma 2 implies that \(x = 0\). Hence, every \((f, g)\)-fixed point satisfies \(y = x = 0\). In view of Theorem 2 it follows that if (78) holds, then \(\epsilon < \epsilon^*_G\), so
\[
\epsilon^*_G \geq \inf_{\substack{y \in [0,1] \\ q_J(y) \leq 1}} y \cdot g(1, q(y), y) \tag{80}
\]
which with (77) completes the proof when \(P_0 = 0\) or \(\lambda_J(0) > 0\).

2) If \(P_0 > 0\) and \(\lambda_J(0) = 0\), it is not true in general that for every fixed point \((x, y)\), \(y = 0\) implies \(x = 0\). However, if in addition to (78),
\[
\epsilon < \frac{1}{P_0} \cdot \inf_{[0,1]} \frac{x}{\lambda_L (1 - \rho_L (1 - x))}, \tag{81}
\]
and \(y = 0\) for some fixed point \((x, y)\), then \(x = 0\). To see this, assume to the contrary that \(x > 0\). In view of (14a) and (81) it follows that
\[
x = f(\epsilon, x, 0) = \epsilon \cdot P_0 \cdot \lambda_L (1 - \rho_L (1 - x)) < x \tag{82}
\]
in contradiction; hence, if (78) and (81) hold, \( x = 0 \) thus \( \epsilon < \epsilon^*_G \). This means that

\[
\epsilon^*_G \geq \min \left\{ \inf_{y \in [0,1]} \frac{y}{g(1,q(y),y)}, \frac{1}{P_0} \cdot \inf_{(0,1]} \frac{x}{\lambda_L(1 - \rho_L(1 - x))} \right\}. \quad (83)
\]

To complete the proof, we must show that when \( P_0 > 0 \) and \( \lambda_J(0) = 0 \), then

\[
\epsilon^*_G \leq \min \left\{ \inf_{y \in [0,1]} \frac{y}{g(1,q(y),y)}, \frac{1}{P_0} \cdot \inf_{(0,1]} \frac{x}{\lambda_L(1 - \rho_L(1 - x))} \right\}. \quad (84)
\]

If

\[
\inf_{y \in [0,1]} \frac{y}{g(1,q(y),y)} \leq \frac{1}{P_0} \inf_{(0,1]} \frac{y}{\lambda_L(1 - \rho_L(1 - y))},
\]

then (84) follows immediately from (77); hence we can assume that

\[
\inf_{y \in [0,1]} \frac{y}{g(1,q(y),y)} > \frac{1}{P_0} \inf_{(0,1]} \frac{y}{\lambda_L(1 - \rho_L(1 - y))}. \quad (85)
\]

Let

\[
\inf_{y \in [0,1]} \frac{y}{g(1,q(y),y)} > \epsilon > \frac{1}{P_0} \inf_{(0,1]} \frac{y}{\lambda_L(1 - \rho_L(1 - y))}, \quad (86)
\]

and let \( x_0 \in (0,1] \), such that \( x_0 = \epsilon \cdot P_0 \cdot \lambda_L (1 - \rho_L (1 - x_0)) \). Since \( \lambda_J(0) = 0 \), it follows that \( (x_0,0) \) is a fixed point with \( x_0 > 0 \), thus \( \epsilon > \epsilon^*_G \). Since this is true for every \( \epsilon > \frac{1}{P_0} \inf_{(0,1]} \frac{y}{\lambda_L(1 - \rho_L(1 - y))} \), then \( \epsilon^*_G \leq \frac{1}{P_0} \inf_{(0,1]} \frac{y}{\lambda_L(1 - \rho_L(1 - y))} \). In view of (85), it follows that (84) holds. This completes the proof for the \( P_0 > 0 \) and \( \lambda_J(0) = 0 \) case.

**APPENDIX E**

**PROOF OF LEMMA 6**

In view of (8),

\[
\epsilon_L = \inf_{(0,1]} \frac{x}{\lambda_L(1 - \rho_L(1 - x))}
\]

\[
= \inf_{(0,1]} \frac{x}{1 - \sum_{i=2}^4 \rho_i (1 - x)^{i-1}}.
\]

Since \( \rho_2 = 1 - \rho_3 - \rho_4 \), we get

\[
\epsilon_L = \inf_{(0,1]} \frac{x}{(1 + \rho_3 + 2\rho_4)x - (\rho_3 + 3\rho_4)x^2 + \rho_4x^3}
\]

\[
= \frac{1}{\sup_{(0,1]} \left[ (1 + \rho_3 + 2\rho_4) - (\rho_3 + 3\rho_4)x + \rho_4x^2 \right]}
\]

\[
= \frac{1}{1 + \rho_3 + 2\rho_4}.
\]
In view of Definition 4, we have

\[ h_\epsilon(x) = \epsilon \left[ (1 + \rho_3 + 2\rho_4)x - (\rho_3 + 3\rho_4)x^2 + \rho_4 x^3 \right] - x \]

\[ = x \left( \epsilon \left[ (1 + \rho_3 + 2\rho_4) - (\rho_3 + 3\rho_4)x + \rho_4 x^2 \right] - 1 \right), \quad (87) \]

thus \( x_\epsilon(\epsilon) \) is a root of \( \rho_4 x^2 - (\rho_3 + 3\rho_4)x + (1 + \rho_3 + 2\rho_4) - \frac{1}{\epsilon} \) in \((0, 1)\). If \( \rho_4 = 0 \), then \((39)\) implies \( \rho_3 = \frac{1}{\epsilon_L} - 1 > \frac{1}{\epsilon} - 1 \), so the solution given by \((40)\) is in \((0, 1)\). If \( \rho_4 > 0 \), then the quadratic formula implies

\[ x_\epsilon(\epsilon) = \frac{\rho_3 + 3\rho_4 \pm \sqrt{(\rho_3 + \rho_4)^2 + 4\rho_4 \left( \frac{\epsilon}{\epsilon - 1} \right)}}{2\rho_4}. \quad (88) \]

Note that \( ' + ' \) option violates the constraint \( x_\epsilon(\epsilon) \leq 1 \). Moreover, since \( \epsilon > \epsilon_L = \frac{1}{1 + \rho_3 + 2\rho_4} \), then

\[ \sqrt{(\rho_3 + \rho_4)^2 + 4\rho_4 \left( \frac{\epsilon}{\epsilon - 1} \right)} \leq \sqrt{\rho_3^2 + 6\rho_3\rho_4 + 9\rho_4^2} = \rho_3 + 3\rho_4; \]

hence \( x_\epsilon(\epsilon) \) as given in \((41)\) is strictly positive. Furthermore, we have \( \sqrt{(\rho_3 + \rho_4)^2 + 4\rho_4 \left( \frac{\epsilon}{\epsilon - 1} \right)} \geq \rho_3 + \rho_4 \), so \( x_\epsilon(\epsilon) \leq 1 \).

Finally, since \( \lambda_L(x) = x \), then by definition \( \frac{x_\epsilon(\epsilon)}{\epsilon} = 1 - \rho_L(1 - x_\epsilon(\epsilon)) \), which yields \( a_\epsilon(\epsilon) = \left( \frac{x_\epsilon(\epsilon)}{\epsilon} \right)^2 \), and completes the proof.

**APPENDIX F**

**PROOF OF LEMMA 8**

To prove Lemma 8 we need the following lemma.

**Lemma 11:** A scheduling scheme is valid if and only if, \( \epsilon_{loc}(y_l) < \epsilon_L \), for some iteration \( l \).

**Proof:** Recall the definition of the local threshold,

\[ \epsilon_L = \sup \{ \epsilon : x = \epsilon \lambda_L(1 - \rho_L(1 - x)) \text{ has no solution in } (0, 1) \}, \quad (89) \]

and let \( x = \lim_{l \to \infty} x_l \) and \( y = \lim_{l \to \infty} y_l \). Since under every scheduling scheme \( y_l \) is monotonically non-increasing in \( l \), then in view of \((54)\),

\[ \exists l \in \mathbb{N}, \quad \epsilon_{loc}(y_l) < \epsilon_L \]

\[ \Downarrow \]

\[ \epsilon_{loc}(y) < \epsilon_L \]

\[ \Downarrow \]

\[ x = \epsilon_{loc}(y)\lambda_L(1 - \rho_L(1 - x)) \text{ has no solution for } x \in (0, 1] \]

\[ \Downarrow \]

\[ \lim_{l \to \infty} x_l = 0. \]
Let \((\Lambda_L, \Lambda_J, \Omega_L, \Omega_J)\) be degree-distribution polynomials, and let \(\epsilon \leq \epsilon^*_G(\Lambda_L, \Lambda_J, \Omega_L, \Omega_J)\).

Let \((x, y) = \lim_{l \to \infty} (x_l, y_l)\). Assume in contradiction that \(\epsilon_{\text{loc}}(y) \geq \epsilon_L\). Since \(\eta > 0\), letting \(l \to \infty\) in \(55\) implies that \((x, y)\) is a non-trivial \((f, g)\)-fixed point. However, in view of Theorem 2, if \(\epsilon \leq \epsilon^*_G(\Lambda_L, \Lambda_J, \Omega_L, \Omega_J)\), every \((f, g)\)-fixed point is the trivial point, in contradiction. Thus, \(\epsilon_{\text{loc}}(y) < \epsilon_L\) which, due to Lemma 11 completes the proof.

**Appendix G**

**Proof of Lemma 9**

Let \(\{l_k^{(1)}\}_{k=1}^{N_{JI}^{(1)}}\) and \(\{l_k^{(2)}\}_{k=1}^{N_{JI}^{(2)}}\) be the joint-update iterations of the scheduling scheme described in \(56d\) and in some arbitrary valid scheduling scheme, receptively. We need to show that \(N_{JI}^{(1)} \leq N_{JI}^{(2)}\). To proceed we need the next lemma:

**Lemma 12:**

\[
y_{l_k^{(1)}} \leq y_{l_k^{(2)}}, \quad 1 \leq k \leq \min\left(N_{JI}^{(1)}, N_{JI}^{(2)}\right). \tag{90}
\]

**Proof:** By induction on \(1 \leq k \leq \min\left(N_{JI}^{(1)}, N_{JI}^{(2)}\right)\). In the first joint update, we have \(y_{l_k^{(1)}} = 1 = y_{l_k^{(2)}}\), so \(90\) holds for \(k = 1\). Assume correctness for some joint update \(k < \min\left(N_{JI}^{(1)}, N_{JI}^{(2)}\right)\), and consider update \(k + 1\). In view of Lemma 1, \(56b\)-\(56d\), \(58\) and the induction assumption,

\[
y_{l_{k+1}^{(1)}} = g\left(\epsilon, x_{l_{k+1}^{(1)}}, y_{l_{k+1}^{(1)}}\right) = g\left(\epsilon, x_k, y_{l_k^{(1)}}, y_{l_k^{(1)}}\right) \leq g\left(\epsilon, x_k, y_{l_k^{(2)}}, y_{l_k^{(2)}}\right) = y_{l_{k+1}^{(2)}},
\]

which by induction, completes the proof.

Assume, on the contrary, that \(N_{JI}^{(1)} > N_{JI}^{(2)}\). Let

\[
\epsilon_k^{(1)} = \epsilon_{\text{loc}}\left(y_{l_k^{(1)}}\right), \quad 1 \leq k \leq N_{JI}^{(1)},
\]

\[
\epsilon_k^{(2)} = \epsilon_{\text{loc}}\left(y_{l_k^{(2)}}\right), \quad 1 \leq k \leq N_{JI}^{(2)}. \tag{91}
\]

Lemma 12 implies that \(y_{l_k^{(1)}} \leq y_{l_k^{(2)}}\), for every \(k \in \{1, \ldots, N_{JI}^{(2)}\}\). Hence,

\[
\epsilon_k^{(1)} \leq \epsilon_k^{(2)}, \quad \forall k \in \{1, \ldots, N_{JI}^{(2)}\}. \tag{92}
\]
Since $N^{(1)}_{JI} > N^{(2)}_{JI}$, (57) and (92) imply
\[ \epsilon^{(2)}_{N^{(2)}_{JI}} \geq \epsilon^{(1)}_{N^{(2)}_{JI}} \geq \epsilon^{(1)}_{N^{(1)}_{JI}} - 1 \geq \epsilon_L, \]
which, in view of Lemma 11 yields that the scheduling scheme indexed by \( \left\{ l^{(2)}_{k} \right\}_{k=1}^{N^{(2)}_{JI}} \) is not valid, in contradiction. Thus, $N^{(1)}_{JI} \leq N^{(2)}_{JI}$.

**APPENDIX H**

**PROOF OF LEMMA 10**

In view of Definition 4, let $h^{(k)}_{\epsilon}(x) = \epsilon \lambda^{(k)}(1 - \rho^{(k)}(1 - x)) - x$. Eq. (59) yields
\[ h_{\epsilon}(x) = \lim_{k \to \infty} h^{(k)}_{\epsilon}(x) = \begin{cases} \left( \frac{\epsilon}{\epsilon_L} - 1 \right) x & 0 \leq x \leq \epsilon_L \\ \epsilon - x & \epsilon_L \leq x \leq \epsilon \end{cases} \]

For every $k \in \mathbb{N}$, and $x \in (\epsilon, 1]$,
\[ h^{(k)}_{\epsilon}(x) = \epsilon \lambda^{(k)}(1 - \rho^{(k)}(1 - x)) - x \]
\[ \leq \epsilon - x \]
\[ < 0, \]
Thus
\[ x^{(k)}_{s}(\epsilon) \leq \epsilon, \quad \forall k \in \mathbb{N}. \]

In addition, for every $0 < a < \epsilon$ there exists $K_0$ such that
\[ h^{(k)}_{\epsilon}(\epsilon - a) > 0, \quad \forall k \geq K_0, \]
so $x^{(k)}_{s}(\epsilon) \geq \epsilon - a$, for every $k \geq K_0$; hence,
\[ \liminf_{k \to \infty} x^{(k)}_{s}(\epsilon) \geq \epsilon. \]
Combining (95) and (96) implies that $\lim_{k \to \infty} x^{(k)}_{s}(\epsilon)$ exists, and completes the proof.

**REFERENCES**

[1] M. Blaum and S. Hetzler, “Integrated interleaved codes as locally recoverable codes: properties and performance,” Int. J. Inf. Coding Theory, vol. 3, no. 4, pp. 324–344, Jan. 2016.

[2] Y. Cassuto, E. Hemo, S. Puchinger and M. Bossert, “Multi-block interleaved codes for local and global read access”, Proc. 2017 IEEE International Symposium on Information Theory, pp. 1758–1762, Aachen, Germany, June 2017.
[3] C. Di, D. Proietti, E. Teletar, T. J. Richardson, and R. L. Urbanke, “Finite-length analysis of low-density parity-check codes on the binary erasure channel”, *IEEE Trans. on Inf. Theory*, vol. 48, no. 6, pp. 1570–1579, Aug. 2002.

[4] D. Divsalar, H. Jin, and R. J. McEliece, “Coding theorems for turbo-like codes,” *Proc. 36th Allerton Conf. on Communication, Control, and Computing*, pp. 201-210, Allerton, Illinois, Sept. 1998.

[5] R. Gallager, “Low-density parity-check codes,” *IRE Trans. on Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[6] M. Hassner, K. Abdel-Ghaar, A. Patel, R. Koetter, and B. Trager, “Integrated interleaving a novel ECC architecture,” *IEEE Trans. Magn.*, vol. 37, no. 3, pp. 773-775, Feb. 2001.

[7] J. Han and L. A. Lastras-Montano, “Reliable memories with subline accesses,” *Proc. 2007 IEEE International Symposium on Information Theory*, pp. 2531–2535, Nice, France, June 2007.

[8] H. Jin A. Khandekar and R. J. McEliece, “Irregular repeat-accumulate codes,” *Proc. 2nd Int. Conf. Turbo Codes and Related Topics*, pp. 18, Brest, France, Sep. 2000.

[9] S. Kudekar, T. J. Richardson, and R. L. Urbanke, “Threshold saturation via coupling: why convolutional LDPC ensembles perform so well over the BEC,” *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 803-834, Feb. 2011.

[10] J. Li, K. Liu, S. Lin, K. Abdel-Ghaffar, “Reed-solomon based nonbinary globally coupled LDPC codes: Correction of random errors and bursts of erasures”, *Proc. 2017 IEEE International Symposium on Information Theory*, pp. 381–385, Aachen, Germany, June 2017.

[11] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman, “Efficient erasure correcting codes,” *IEEE Trans. on Inf. Theory*, vol. 47, no. 2, pp. 569-584, Feb. 2001.

[12] P. Oswald and M. A. Shokrollahi, “Capacity-achieving sequences for the erasure channel, *IEEE Trans. on Inf. Theory*, vol. 48, no. 12, pp. 3017-3028, Dec. 2002.

[13] H. D. Pfister and I. Sason, “AccumulateRepeatAccumulate codes: capacity-achieving ensembles of systematic codes for the erasure channel with bounded complexity”, *IEEE Trans. on Inf. Theory*, vol. 53, no. 6, pp. 2088–2115, June. 2007.

[14] P. Razaghi and W. Yu, “Bilayer low-density parity-check codes for decode-and-forward in relay channels”, *IEEE Trans. on Inf. Theory*, vol. 53, no. 10, pp. 3723-3739, Sep. 2007.

[15] T. Richardson and R. Urbanke, *Modern Coding Theory*, New York, NY, USA: Cambridge Univ. Press, 2008.

[16] M. A. Shokrollahi, “New sequences of linear time erasure codes approaching the channel capacity,” in *Proc. 13th International Symposium on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes*, vol. 1719, pp. 65-76, Nov. 1999.