Design and Decoding of Irregular LDPC Codes Based on Discrete Message Passing

Michael Meidlinger, Gerald Matz, *Senior Member, IEEE,* and Andreas Burg, *Member, IEEE*

***Abstract*— We consider discrete message passing (MP) decoding of low-density parity check (LDPC) codes based on information-optimal symmetric look-up table (LUT). A link between discrete message labels and the associated log-likelihood ratio values (defined in terms of density evolution distributions) is established. This link gives rise to an algebraic structure on the message labels and leads to an interpretation of LUT decoding as a form of quantized belief propagation. We then exploit the algebraic structure for low-complexity LUT decoder designs. Our LUT decoding framework is the first to also apply to irregular LDPC codes by taking into account the degree distribution in a joint LUT design. We exploit the relation between LUT decoding and belief propagation to obtain stability conditions and irregular LDPC code designs optimized for LUT decoding. The resulting decoders outperform floating-point precision min-sum decoders at LUT resolutions as low as 3 bits for regular codes and 4 bits for irregular codes.**

***Index Terms*— Low-density parity check (LDPC) decoder, code design, message passing (MP), quantization.**

I. Introduction

A. Background

LOW-DENSITY parity check (LDPC) codes, originally introduced in [3], are a powerful class of error-correcting codes. Irregular LDPC codes [4], [5] outperform the best known turbo codes and can approach channel capacity using iterative message passing (MP) decoding. Due to their excellent error rate performance LDPC codes are widely used, e.g., in flash storage [6], [7] and in communication systems such as DVB-S2 [8], WiMAX [9], and 10GBASE-T [10].

In view of the vast range of applications, recent research aims at devising LDPC decoders that are well suited for practical digital hardware implementations [11], [12]. Most of these schemes are variants or approximations of belief propagation (BP) and rely on fixed-point representations of log-likelihood ratios (LLRs) with a resolution of 5 to 7 bits. Low resolution reduces decoder complexity but usually deteriorates error rate performance [12]–[15]. An alternative approach is *finite alphabet* LDPC decoding [16]–[22], often also referred to as look-up table (LUT) decoding. Rather than performing conventional BP or min-sum (MS) decoding with quantized message representations, the message updates are performed via dedicated LUTs (discrete finite mappings). In LUT decoding, algebraic structure and the probabilistic message interpretation is sacrificed in exchange for a drastic reduction in message bit-width and an improved error rate performance. The LUT design has been based on deterministic information-optimal quantizers [16], [21], on the information bottleneck [23] (e.g., [17]–[19]), and on error pattern correction [20]. However, the applicability of existing work is restricted to regular LDPC codes and codes with low node degrees. Further related work [24], applicable to irregular LDPC codes, has appeared after the initial submission of this manuscript. Our own previous work [1], [2] focused on simplifying LUT decoders and making them more suitable for practical implementations, giving rise to the decoder architectures in [25], [26].

B. Contributions and Paper Outline

In this paper, we provide a systematic, coherent, and general framework for LUT decoding of regular and irregular LDPC codes. Such a framework has been lacking in the existing literature. Using a dedicated code design, irregular LDPC codes are shown to perform close to capacity even with low-resolution LUT decoding.

After reviewing basic concepts in Section II, we begin our development in Section III by introducing a novel symmetry concept for discrete message passing (MP) using arbitrary message alphabets. Exploiting this symmetry, we use density evolution (DE) arguments (see [27]) to derive a relation between the discrete messages in LUT decoders and the (real-valued) LLRs in BP. We then use this relation to establish an algebraic structure on the discrete messages and we show how LUT decoding corresponds to BP augmented with a projection of outgoing messages onto finite sets. In Section IV we use the symmetries and the algebraic structure from Section III to derive low-complexity LUT decoders based on imposing symmetry in the information-maximizing quantizer design from [28]. Furthermore, we propose a hybrid min-LUT decoding algorithm that uses LUT updates and the algebraic structure of the message labels.

In Section V, we extend LUT decoding to irregular LDPC codes. Exploiting the analogy to BP, we derive an asymptotic

Manuscript received August 22, 2018; revised January 29, 2019, April 26, 2019, July 1, 2019, and August 23, 2019; accepted September 9, 2019. Date of publication September 27, 2019; date of current version March 18, 2020. This research has been supported by WWTF Grants ICT12-054, ICT15-119, and NXT17-013. This article was presented in part at the Asilomar SSC 2015 [1] and in part at the IEEE SPAWC 2017 [2]. The associate editor coordinating the review of this article and approving it for publication was Q. Huang. *(Corresponding author: Michael Meidlinger.)*

M. Meidlinger is with Thales Austria GmbH, 1200 Vienna, Austria (e-mail: michael@meidlinger.info).

G. Matz is with the Institute of Telecommunications, Vienna University of Technology, 1040 Vienna, Austria.

A. Burg is with EPFL, 1015 Lausanne, Switzerland.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCOMM.2019.2944159

stability condition for LUT decoding in the spirit of [5]. The stability condition explains why existing LDPC degree distributions that are optimized for BP perform poorly with LUT decoding. This motivates us to propose degree distributions optimized for LUT decoding. We demonstrate numerically that irregular LDPC codes generated from these degree distributions perform better than existing LDPC codes under LUT decoding.

In Section VI, we present extensive numerical results and compare different LUT-based decoders in terms of error rate and DE thresholds. A performance-complexity tradeoff is established that can be used to design a decoder tailored to specific error rate requirements and implementation constraints.

We note that we also have developed a comprehensive C++/Matlab software framework for the design, implementation, and evaluation of our LUT decoders for LDPC codes. The software is publicly available and can be accessed via the link http://www.nt.tuwien.ac.at/UNFOLD.

II. LDPC Basics

A. Message Passing

We consider ensembles of binary LDPC codes defined on the finite field $\{-1, 1\}$ of size two, with 1 representing the neutral element (see [29, Section 4.1.1]). A particular code is represented by a factor graph [30] consisting of N variable nodes (VNs) and M check nodes (CNs). The edges between VNs and CNs are characterized by the degree distributions [29, p. 79]

$$\lambda(\xi) = \sum_{i \in \mathcal{D}_v} \lambda_i \, \xi^{i-1}, \quad \rho(\xi) = \sum_{j \in \mathcal{D}_c} \rho_j \, \xi^{j-1}. \tag{1}$$

Here λ_i, $i \in \mathcal{D}_v$, is the probability that an edge is incident to a VN of degree i, and ρ_j, $j \in \mathcal{D}_c$, is the probability that an edge is incident to a CN of degree j, with $\mathcal{D}_v$ and $\mathcal{D}_c$ denoting the VN and CN degree sets, respectively. In a regular (d_v, d_c) LDPC code, all VNs have degree d_v and all CNs have degree d_c (i.e., $\lambda(\xi) = \xi^{d_v - 1}$ and $\rho(\xi) = \xi^{d_c - 1}$). LDPC codes are typically decoded using MP algorithms, where messages are exchanged between VNs and CNs over the course of several decoding iterations [27]. Figure 1 illustrates the message updates in the factor graph. In iteration ℓ, the message from VN n (with degree i) to an adjacent CN m is computed via a mapping $\Phi_i^{(\ell)} : \mathcal{L} \times \overline{\mathcal{M}}_\ell^{i-1} \to \mathcal{M}_\ell$ ($\overline{\mathcal{M}}_\ell$, $\mathcal{M}_\ell$, and $\mathcal{L}$ are the message and LLR alphabets; bars designate messages, updates, and sets at the CNs) as

$$\mu_{n \to m} = \Phi_i^{(\ell)}\big(L_n, \bar{\boldsymbol{\mu}}_{n \setminus m}\big). \tag{2}$$

Here, $L_n \in \mathcal{L}$ denotes the channel LLR at VN n and $\bar{\boldsymbol{\mu}}_{n \setminus m} \in \overline{\mathcal{M}}_\ell^{i-1}$ is the vector of incoming messages from all adjacent CNs except m. Similarly, the message in iteration ℓ from CN m (with degree j) to an adjacent VN n is obtained via the mapping $\bar{\Phi}_j^{(\ell)} : \mathcal{M}_\ell^{j-1} \to \overline{\mathcal{M}}_{\ell+1}$,

$$\bar{\mu}_{m \to n} = \bar{\Phi}_j^{(\ell)}(\boldsymbol{\mu}_{m \setminus n}), \tag{3}$$

where $\boldsymbol{\mu}_{m \setminus n} \in \mathcal{M}_\ell^{j-1}$ is the vector of incoming messages from all adjacent VNs except n.

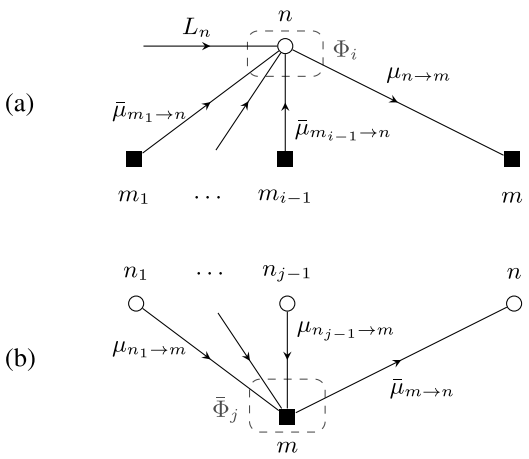

Fig. 1. Illustration of (a) VN update and (b) CN update in an LDPC factor graph.

After a total of $\mathscr{L}$ iterations, a mapping $\Psi_i : \mathcal{L} \times \overline{\mathcal{M}}_{\mathscr{L}}^{i} \to \{-1, 1\}$ is used to compute the final estimate for a code bit x_n at the associated VN of degree i using the incoming CN messages and the channel LLR as

$$\hat{x}_n = \Psi_i(L_n, \bar{\boldsymbol{\mu}}_n).$$

Designing a decoder for a given LDPC code amounts to determining the VN updates $\Phi_i^{(\ell)}$, the CN updates $\bar{\Phi}_j^{(\ell)}$, and the bit decisions Ψ_i for all VN degrees $i \in \mathcal{D}_v$, CN degrees $j \in \mathcal{D}_c$, and iterations $\ell \in \{1, \ldots, \mathcal{L}\}$.

For BP, all message alphabets equal the set of reals and the message updates (which are identical for all iterations) read

$$\Phi_i^{\mathrm{BP}}(L, \bar{\boldsymbol{\mu}}) = L + \sum_{m=1}^{i-1} \bar{\mu}_m,$$
$$\bar{\Phi}_j^{\mathrm{BP}}(\boldsymbol{\mu}) = 2 \operatorname{atanh}\Bigg(\prod_{n=1}^{j-1} \tanh\Big(\frac{\mu_n}{2}\Big)\Bigg). \tag{4}$$

The expression on the right-hand side of (4) is known as the box-plus operation [31]. For the MS algorithm, (4) is approximated by

$$\bar{\Phi}_j^{\mathrm{MS}}(\boldsymbol{\mu}) = \operatorname{par}\boldsymbol{\mu} \cdot \min|\boldsymbol{\mu}|, \tag{5}$$

where $\operatorname{par} \boldsymbol{x} = \prod_i \operatorname{sgn}(x_i)$ and $|\boldsymbol{x}|$ respectively denote the parity and the element-wise magnitude of the vector $\boldsymbol{x}$. The bit decision Ψ_i is based on the sign of the a posteriori LLRs,

$$\Psi_i^{\mathrm{MS/BP}}(L, \bar{\boldsymbol{\mu}}) = \operatorname{sgn}\Bigg(L + \sum_{m=1}^{i} \bar{\mu}_m \Bigg).$$

B. Density Evolution and Symmetry Conditions

In what follows, we use sans serif letters for random variables and denote their (conditional) distribution generically as $p(\cdot)$. In particular, $p_{\mathsf{L}|\mathsf{x}}(L|x)$ denotes the binary input channel, which we assume to be symmetric (here, L is the LLR for input bit x). Furthermore, $\boldsymbol{x} \odot \boldsymbol{y}$ denotes the element-wise product of two vectors.

Density evolution (DE) was established as a method of investigating the asymptotic performance of LDPC codes [27] and (in its initial form) relies on MP schemes that are symmetric in the sense

$$p_{\mathsf{L}|\mathsf{x}}(L|x) = p_{\mathsf{L}|\mathsf{x}}(-L|-x), \tag{6}$$

$$\Phi_i^{(\ell)}(-L, -\bar{\boldsymbol{\mu}}) = -\Phi_i^{(\ell)}(L, \bar{\boldsymbol{\mu}}), \tag{7}$$

$$\bar{\Phi}_j^{(\ell)}(\boldsymbol{b} \odot \boldsymbol{\mu}) = \bar{\Phi}_j^{(\ell)}(\boldsymbol{\mu}) \mathrm{par}\, \boldsymbol{b}, \tag{8}$$

for all vectors $\boldsymbol{b} \in \{-1, 1\}^{j-1}$, iterations ℓ, and degrees $i \in \mathcal{D}_v$, $j \in \mathcal{D}_c$.

Using these symmetry conditions and restricting to codes with cycle-free graphs, the VN→CN and the CN→VN messages are conditionally independent and identically distributed (iid) random variables. With $n(e)$ denoting the VN incident to a particular edge $e \in \mathcal{E}$, we specifically have

$$p_{\boldsymbol{\mu}|\boldsymbol{x}}^{(\ell)}(\boldsymbol{\mu}|\boldsymbol{x}) = \prod_{e \in \mathcal{E}} p_{\mu|\mathsf{x}}^{(\ell)}(\mu_e|x_{n(e)}),$$

with symmetric distributions

$$p_{\mu|\mathsf{x}}^{(\ell)}(\mu_e| - x_{n(e)}) = p_{\mu|\mathsf{x}}^{(\ell)}(-\mu_e|x_{n(e)}). \tag{9}$$

DE then tracks the evolution of the message densities (9) for $\ell \to \infty$ to determine whether the error probability tends to zero for a given code degree distribution (1) and channel (6). For channels characterized by a single parameter (e.g., the noise variance in AWGN channels), the *decoding threshold* is defined as the worst channel parameter such that DE converges to zero error probability.

Another condition (cf. [5, Definition 1]) requires that the symmetric conditional message distributions satisfy

$$\mu = \log \frac{p(\mu|x{=}+1)}{p(\mu|x{=}-1)} \iff p(\mu|x{=}+1) = \mathrm{e}^{\mu} p(\mu|x{=}-1). \tag{10}$$

This connection between the conditional message distributions is referred to as *LLR consistency* since it implies that the LLR associated with a conditional message distribution equals the LLR value itself.

III. Symmetric Discrete MP

Even though vital to MP and DE, symmetry concepts have not been studied in a systematic manner in prior work on LUT decoding (though particular symmetry conditions have been adopted in an ad hoc manner [17], [20]). We next derive a novel general framework for discrete DE that links LUT decoding to BP. We first extend the symmetry conditions from Section II-B to discrete message labels and then associate the labels to LLR values using the message probability mass functions (pmfs) from DE. We use this connection in several ways:

i) We establish an algebraic structure on the message label set, which we exploit in Section IV for reduced-complexity LUT design as well as for the derivation of the min-LUT algorithm.
ii) We show that symmetric discrete MP can be interpreted as quantized BP, where LLRs are projected onto a finite set after each message update.
iii) We extend the asymptotic stability analysis for BP decoding [5] to LUT decoding (see Section V). We exploit the stability result to design irregular LDPC codes optimized for LUT decoding.

A. Message Labels, LLRs, and Symmetry

Let y denote a generic message label from a finite message set $\mathcal{Y} = \{y_0, y_1, \ldots, y_{M-1}\}$ with conditional probability mass function (pmf) $p_{\mathsf{y}|\mathsf{x}}(y|x)$. The set $\mathcal{Y}$ is a generic placeholder for $\mathcal{L}$, $\mathcal{M}_\ell$, $\overline{\mathcal{M}}_\ell$, and for the product sets $\mathcal{L} \times \overline{\mathcal{M}}_\ell^{i-1}$ and $\mathcal{M}_\ell^{j-1}$. Note that the elements of $\mathcal{Y}$ are totally arbitrary and could be integers or (in hardware implementations practically more important) bit labels. Henceforth, we assume that all message sets have even cardinality. We define the LLR value associated to label y in terms of its conditional label pmf as

$$\upsilon \triangleq \Lambda(y) = \log \frac{p_{\mathsf{y}|\mathsf{x}}(y|+1)}{p_{\mathsf{y}|\mathsf{x}}(y|-1)}. \tag{11}$$

Without loss of generality, we restrict ourselves to the case where (11) is bijective. If multiple labels were associated to the same non-zero LLR, the labels can be merged without information loss [28]; since we enforce symmetry in our design, the cardinality of the message set remains even after label merging. The exception is multiple labels with LLR equal to 0: we merge half of these labels into a symbol y_k and the other half into a symbol y_l (symmetry is maintained by ensuring that no label y is merged with its inverse $\neg y$ defined below). We then use the convention $\Lambda(y_k) = 0^-$, $\Lambda(y_l) = 0^+$, with sign change and order relation for 0^- and 0^+ defined by $-0^+ = 0^-$ and $0^- < 0^+$. Since $\Lambda(\cdot)$ is bijective with inverse $\Lambda^{-1}(\cdot)$, we have

$$p_{\Lambda(\mathsf{y})|\mathsf{x}}(\upsilon|x) = p_{\mathsf{y}|\mathsf{x}}(\Lambda^{-1}(\upsilon)|x), \tag{12}$$

which together with (11) implies that the distribution of $\upsilon = \Lambda(\mathrm{y})$ is LLR consistent (see (10)).

We next introduce a general definition of symmetry for discrete conditional pmfs. To this end, we generalize algebraic sign changes via involutions (i.e., self-inverse permutations) $\neg$ on $\mathcal{Y}$,

$$\neg(\neg y) = y, \quad y \in \mathcal{Y}.$$

A conditional pmf $p_{\mathsf{y}|\mathsf{x}}(y|x)$, $y \in \mathcal{Y}$, $x \in \{-1, 1\}$, is said to be symmetric if there exists an involution $\neg : \mathcal{Y} \to \mathcal{Y}$ such that for any $y \in \mathcal{Y}$

$$p_{\mathsf{y}|\mathsf{x}}(y|+1) = p_{\mathsf{y}|\mathsf{x}}(\neg y|-1).$$

For symmetric label pmfs, label inversion amounts to sign inversion of the LLR,

$$\Lambda(\neg y) = -\Lambda(y). \tag{13}$$

The conventional ordering $<$ of the LLRs defines an ordering $\prec$ of the labels,

$$\Lambda(y_k) < \Lambda(y_l) \iff y_k \prec y_l. \tag{14}$$

Without loss of generality, we assume that the labels are sorted according to increasing LLRs,

$$\begin{aligned} y_0 \prec y_1 \prec \cdots \prec y_{M-1}, \\ \Lambda(y_0) < \Lambda(y_1) < \cdots < \Lambda(y_{M-1}). \end{aligned} \tag{15}$$

Using this ordering, a suitable involution is given by

$$\neg y_k = y_{M-1-k}. \tag{16}$$

With these definitions, all statements from Section II-B carry over to the discrete labels with message sign change ('$-$') replaced by label involution ('$\neg$'). Let $\neg_{\boldsymbol{b}}\boldsymbol{\mu}$ denote the element-wise involution of a discrete message $\boldsymbol{\mu}$, i.e. $(\neg_{\boldsymbol{b}}\boldsymbol{\mu})_i = \neg\mu_i$ if $b_i = -1$ and $(\neg_{\boldsymbol{b}}\boldsymbol{\mu})_i = \mu_i$ if $b_i = +1$. Then, the symmetries (6) to (8) generalize to LUT-based (discrete) MP as

$$p_{\mathsf{L}|\mathsf{x}}(L|x) = p_{\mathsf{L}|\mathsf{x}}(\neg L|-x), \tag{17}$$

$$\bar{\Phi}_j^{(\ell)}(\neg_{\boldsymbol{b}}\boldsymbol{\mu}) = \neg_{\mathrm{par}\boldsymbol{b}}\bar{\Phi}_j^{(\ell)}(\boldsymbol{\mu}), \tag{18}$$

$$\Phi_i^{(\ell)}(\neg L, \neg\bar{\boldsymbol{\mu}}) = \neg\Phi_i^{(\ell)}(L, \bar{\boldsymbol{\mu}}). \tag{19}$$

These conditions imply that throughout the decoding iterations the VN→CN and CN→VN messages are iid with symmetric pmfs $p_{\mu|\mathsf{x}}^{(\ell)}(\mu|x)$ and $p_{\bar{\mu}|\mathsf{x}}^{(\ell)}(\bar{\mu}|x)$, e.g.,

$$p_{\mu|\mathsf{x}}^{(\ell)}(\mu_e|-x_{n(e)}) = p_{\mu|\mathsf{x}}^{(\ell)}(\neg\mu_e|x_{n(e)}) \tag{20}$$

(this relation replaces (9)). The LLRs $\Lambda(\mu)$, $\Lambda(\bar{\mu})$, and $\Lambda(L)$ for the message labels are defined according to (11) using the corresponding message pmfs from DE. We denote the associated LLR sets by $\mathcal{M}_\ell'$, $\overline{\mathcal{M}}_\ell'$ and $\mathcal{L}'$, respectively, e.g., $\mathcal{M}_\ell' = \Lambda(\mathcal{M}_\ell) = \{\Lambda(\mu) : \ \mu \in \mathcal{M}_\ell\}$ (the superscript $'$ indicates that a set consists of LLRs). Note that even when the label sets $\mathcal{M}_\ell$ and $\overline{\mathcal{M}}_\ell$ remain constant, the label distributions and hence the associated LLRs change over the course of the iterations. While LUT decoder implementations only deal with the labels, the pmf evolution is accounted for by using different LUTs over the iterations.

B. Sign-Magnitude Interpretation of Labels

According to (13), label inversion corresponds to a sign inversion of the respective LLR. Using the label ordering (15) and (16) allows us to infer information about the LLR value from the message label. We can further define the sign and magnitude of a label $y \in \mathcal{Y}$ as[1]

$$\mathrm{sign}(y) \triangleq \begin{cases} -1, & \neg y \succ y, \\ 1, & \neg y \prec y, \end{cases} \tag{21}$$

$$\mathrm{abs}(y) \triangleq \begin{cases} y, & \neg y \prec y, \\ \neg y, & \neg y \succ y. \end{cases} \tag{22}$$

With these definitions, the LLR sign $\mathrm{sgn}(\Lambda(y))$ equals the label sign $\mathrm{sign}(y)$. By slight abuse of notation, we denote the parity of a label vector $\boldsymbol{y}$ also by $\mathrm{par}\,\boldsymbol{y} = \prod_i \mathrm{sign}(y_i)$.

[1]Note that the $\mathrm{sign}(\cdot)$ function differs from the usual signum function $\mathrm{sgn}(\cdot)$ in that its domain is a message alphabet.

C. Product Distributions

For the case of discrete message alphabets, the update rules (2) and (3) respectively entail the following discrete DE (actually, probability mass evolution) equations:

$$p_{\mu|\mathsf{x}}^{(\ell)}(\mu|x) = \sum_{(L,\bar{\boldsymbol{\mu}}):\ \Phi^{(\ell)}(L,\bar{\boldsymbol{\mu}})=\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}^{(\ell)}(L,\bar{\boldsymbol{\mu}}|x), \tag{23}$$

$$p_{\bar{\mu}|\mathsf{x}}^{(\ell+1)}(\bar{\mu}|x) = \sum_{\boldsymbol{\mu}:\ \bar{\Phi}^{(\ell)}(\boldsymbol{\mu})=\bar{\mu}} p_{\boldsymbol{\mu}|\mathsf{x}}^{(\ell)}(\boldsymbol{\mu}|x). \tag{24}$$

For iid incoming messages, the pmfs $p_{\mathsf{L},\overline{\boldsymbol{\mu}}|\mathsf{x}}^{(\ell)}(L,\overline{\boldsymbol{\mu}}|x)$ and $p_{\boldsymbol{\mu}|\mathsf{x}}^{(\ell)}(\boldsymbol{\mu}|x)$ have been derived in [16] as

$$p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}^{(\ell)}(L,\bar{\boldsymbol{\mu}}|x) = p_{\mathsf{L}|\mathsf{x}}(L|x)\prod_{m=1}^{i-1} p_{\bar{\mu}|\mathsf{x}}^{(\ell)}(\bar{\mu}_m|x), \tag{25}$$

$$p_{\boldsymbol{\mu}|\mathsf{x}}^{(\ell)}(\boldsymbol{\mu}|x) = 2^{2-j}\sum_{\boldsymbol{x}:\ \mathrm{par}\boldsymbol{x}=x}\prod_{n=1}^{j-1} p_{\mu|\mathsf{x}}^{(\ell)}(\mu_n|x_n), \tag{26}$$

(a sum-product interpretation of these product distributions has been given in [18]). The product pmfs (25) and (26) exhibit a symmetry which will turn out to be useful for LUT design in Section IV.

Lemma 1: For any $\boldsymbol{b} \in \{-1,1\}^{j-1}$ and iid input messages with symmetric pmfs (see (20)), the product distributions (25) and (26) are symmetric in the sense that

$$p_{\mathsf{L},\overline{\boldsymbol{\mu}}|\mathsf{x}}^{(\ell)}(\neg L,\neg\overline{\boldsymbol{\mu}}|x) = p_{\mathsf{L},\overline{\boldsymbol{\mu}}|\mathsf{x}}^{(\ell)}(L,\overline{\boldsymbol{\mu}}|-x), \tag{27}$$

$$p_{\boldsymbol{\mu}|\mathsf{x}}^{(\ell)}(\neg_{\boldsymbol{b}}\boldsymbol{\mu}|x) = p_{\boldsymbol{\mu}|\mathsf{x}}^{(\ell)}(\boldsymbol{\mu}|x\mathrm{par}\,\boldsymbol{b}). \tag{28}$$

Proof: The identity (27) follows directly from (25) using the symmetry (20) of the individual pmfs. To show (28), we use (20) in (26) and the fact that $\mathrm{par}(\boldsymbol{x}\odot\boldsymbol{b}) = \mathrm{par}\,\boldsymbol{x}\mathrm{par}\,\boldsymbol{b}$ such that

$$\begin{aligned} p_{\boldsymbol{\mu}|\mathsf{x}}^{(\ell)}(\neg_{\boldsymbol{b}}\boldsymbol{\mu}|x) &= 2^{2-j}\sum_{\boldsymbol{x}:\ \mathrm{par}\boldsymbol{x}=x}\prod_{n=1}^{j-1} p_{\mu|\mathsf{x}}^{(\ell)}(\mu_n|b_nx_n) \\ &= 2^{2-j}\sum_{\boldsymbol{x}':\ \mathrm{par}(\boldsymbol{x}'\odot\boldsymbol{b})=x}\prod_{n=1}^{j-1} p_{\mu|\mathsf{x}}^{(\ell)}(\mu_n|x_n') \\ &= 2^{2-j}\sum_{\boldsymbol{x}':\ \mathrm{par}\boldsymbol{x}'=x\mathrm{par}\boldsymbol{b}}\prod_{n=1}^{j-1} p_{\mu|\mathsf{x}}^{(\ell)}(\mu_n|x_n') \\ &= p_{\boldsymbol{\mu}|\mathsf{x}}^{(\ell)}(\boldsymbol{\mu}|x\mathrm{par}\,\boldsymbol{b}). \end{aligned}$$

■

D. Relating LUT Decoding and BP

The next result (see Appendix for the proof) establishes an interpretation of LUT decoding as quantized BP. This is accomplished by quantifying the LLRs associated with the message tuples whose pmfs are given by (25) and (26) and linking them to LUT updates with contiguous quantization regions (see Section IV). Note that $\Lambda(L,\bar{\boldsymbol{\mu}})$ and $\Lambda(\boldsymbol{\mu})$ are defined by (11) with $y = (L,\bar{\boldsymbol{\mu}})$ and $y = \boldsymbol{\mu}$, respectively.

Theorem 1: Consider a VN LUT update $\mu = \Phi_i(L,\bar{\boldsymbol{\mu}}) \in \mathcal{M}_{\text{out}}$ with inputs $L \in \mathcal{L}$ and $\bar{\boldsymbol{\mu}} = (\bar{\mu}_1,\ldots,\bar{\mu}_{i-1}) \in \overline{\mathcal{M}}_{\text{in}}^{i-1}$ and a CN LUT update $\bar{\mu} = \bar{\Phi}_j(\boldsymbol{\mu}) \in \overline{\mathcal{M}}_{\text{out}}$ with inputs

TABLE I

ILLUSTRATION OF A LUT $\mu = \Phi(L, \bar{\mu})$ AND THE ASSOCIATED MESSAGE ALGEBRAS FOR A VN WITH $i = 2$, MESSAGE ALPHABETS $\mathcal{L} = \overline{\mathcal{M}} = \mathcal{M} = \{00, 01, 10, 11\}$, ASSOCIATED LLR ALPHABETS $\mathcal{L}' = \Lambda(\mathcal{L}) = \{-3, -1, 1, 3\}$, $\overline{\mathcal{M}}' = \Lambda(\overline{\mathcal{M}}) = \{-4, -2, 2, 4\}$, AND $\mathcal{M}' = \Lambda(\mathcal{M}) = \{-5.5, -1.5, 1.5, 5.5\}$. THE LUT HERE HAS 16 2-bit MEMORY ELEMENTS, AMOUNTING TO A TOTAL SIZE OF 32 bits. HOWEVER, DUE TO THE SYMMETRY $\Phi(\neg(L, \bar{\mu})) = \neg\Phi(L, \bar{\mu}) = \neg\mu$ ONLY HALF OF THE TABLE NEEDS TO BE DESIGNED AND STORED

LUT input					LUT output				
$(L, \bar{\mu})$	$\Lambda(L, \bar{\mu})$	$\text{sign}(L, \bar{\mu})$	$\text{abs}(L, \bar{\mu})$	$\neg(L, \bar{\mu})$	μ	$\Lambda(\mu)$	$\text{sign}(\mu)$	$\text{abs}(\mu)$	$\neg\mu$
(00, 00)	-7	-1	(11,11)	(11,11)					
(00, 01)	-5	-1	(11,10)	(11,10)	00	-5.5	-1	11	11
(01, 00)	-5	-1	(10,11)	(10,11)					
(01, 01)	-3	-1	(10,10)	(10,10)					
(10, 00)	-3	-1	(01,11)	(01,11)					
(00, 10)	-1	-1	(11,01)	(11,01)	01	-1.5	-1	10	10
(10, 01)	-1	-1	(01,10)	(01,10)					
(11, 00)	-1	-1	(00,11)	(00,11)					
(00, 11)	+1	+1	(00,11)	(11,00)					
(01, 10)	+1	+1	(01,10)	(10,01)					
(11, 01)	+1	+1	(11,01)	(00,10)	10	+1.5	+1	10	01
(01, 11)	+3	+1	(01,11)	(10,00)					
(10, 10)	+3	+1	(10,10)	(01,01)					
(10, 11)	+5	+1	(10,11)	(01,00)					
(11, 10)	+5	+1	(11,10)	(00,01)	11	+5.5	+1	11	00
(11, 11)	+7	+1	(11,11)	(00,00)					

$\boldsymbol{\mu} = (\mu_1, \ldots, \mu_{j-1}) \in \mathcal{M}_{\text{in}}^{j-1}$, where all messages are sorted according to (15). The LLRs associated with the VN and CN LUT output satisfy

$$\min_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \Lambda(L, \bar{\boldsymbol{\mu}}) \leq \Lambda(\mu) \leq \max_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \Lambda(L, \bar{\boldsymbol{\mu}}), \quad (29)$$

$$\min_{\boldsymbol{\mu}\in\mathcal{C}_{\bar{\mu}}} \Lambda(\boldsymbol{\mu}) \leq \Lambda(\bar{\mu}) \leq \max_{\boldsymbol{\mu}\in\mathcal{C}_{\bar{\mu}}} \Lambda(\boldsymbol{\mu}). \quad (30)$$

Here, $\mathcal{V}_\mu = \{(L, \bar{\boldsymbol{\mu}}) : \Phi_i(L, \bar{\boldsymbol{\mu}}) = \mu\} \subset \mathcal{L} \times \overline{\mathcal{M}}_{\text{in}}^{i-1}$ and $C_{\bar{\mu}} = \{\boldsymbol{\mu} : \bar{\Phi}_j(\boldsymbol{\mu}) = \bar{\mu}\} \subset \mathcal{M}_{\text{in}}^{j-1}$ are the pre-images of the VN and CN LUT updates, respectively, i.e., all input message tuples that are mapped to the same output message. The LLRs for the tuples $(L, \bar{\boldsymbol{\mu}})$ and $\boldsymbol{\mu}$ are given by

$$\Lambda(L, \bar{\boldsymbol{\mu}}) = \Lambda(L) + \sum_{m=1}^{i-1} \Lambda(\bar{\mu}_m) \quad (31)$$

$$= \Phi_i^{\text{BP}}(\Lambda(L), \Lambda(\bar{\mu}_1), \ldots, \Lambda(\bar{\mu}_{i-1})), \quad (32)$$

$$\Lambda(\boldsymbol{\mu}) = 2\text{atanh}\left(\prod_{n=1}^{j-1} \tanh\left(\frac{\Lambda(\mu_n)}{2}\right)\right) \quad (33)$$

$$= \bar{\Phi}_j^{\text{BP}}\big(\Lambda(\mu_1), \ldots, \Lambda(\mu_{j-1})\big). \quad (34)$$

For a VN LUT that maps contiguous input messages $(L, \bar{\boldsymbol{\mu}}) \in \mathcal{V}_\mu$ to the output message μ, the LLR of μ represents a quantization of the LLRs (and hence of the BP message Φ_i^{BP}) of all $(L, \bar{\boldsymbol{\mu}}) \in \mathcal{V}_\mu$. Similarly, for a contiguous CN LUT the LLR of the output $\bar{\mu}$ associated to the input messages $\boldsymbol{\mu} \in \mathcal{C}_{\bar{\mu}}$ is a quantized version of the LLRs (and hence of the BP message $\bar{\Phi}_j^{\text{BP}}$) of all $\boldsymbol{\mu} \in \mathcal{C}_{\bar{\mu}}$. Thus, LUT decoding essentially amounts to a form of quantized BP. A simple illustrative example with binary message labels is provided in Table I for a VN LUT that is symmetric (e.g., $\Phi(\neg(00, 10)) = \Phi(11, 01) = 10 = \neg\Phi(00, 10) = \neg 01$). For $L = 10$ and $\bar{\mu} = 00$ the LUT update $\mu = \Phi(10, 00) = 01$ with associated LLR $\Lambda(01) = -1.5$ can be viewed as quantized version of the BP update $\Lambda(10, 00) = \Lambda(10) + \Lambda(00) = 1 - 4 = -3$.

We emphasize that BP with LLR updates according to (31) and (33) increases the cardinality of the message sets in each iteration. For example, a (3, 6) LDPC code operating over a binary input additive white Gaussian noise (AWGN) channel at noise level $\sigma = .84$ and with 2-bit channel output quantization [32] entails the LLRs $\mathcal{L}' = \Lambda(\mathcal{L}) = \{-3.39, -0.94, 0.94, 3.39\}$ for the labels $\mathcal{L} = \{0, 1, 2, 3\}$. The unquantized CN update (33) and the subsequent VN update (31) increase the size of the message label sets to 12 and 292, respectively (see [33, Section 4.1.4] for further details). By contrast, LUT updates constrain the size of the output label set and let the decoding progress by changing the LUT updates (and thus the associated LLRs) over the course of the iterations (cf. Figure 2) rather than by growing the label alphabets.

IV. LUT DECODERS

We next consider the LUT design problem, following the approach of [16], [28] to construct LUTs that maximize the local information flow through the code's factor graph with the code bits serving as relevance variables (see [34]). Throughout the rest of the paper, we enforce the VN and CN LUT symmetries (18) and (19) and exploit the algebraic structure from Section III-B to simplify the LUT design, to reduce LUT complexity, and to replace the CN LUTs by an algebraic update rule. We further discuss LUT trees as a means to control LUT size specifically for high-degree nodes.

A. Information-Optimal LUT Design

Consider a joint distribution $p_{\text{x,y}}(x, y)$ on a finite set $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} = \{-1, 1\}$ and $\mathcal{Y} = \{y_0, \ldots, y_{M-1}\}$. Here, $\mathcal{Y}$ denotes a generic set representing tuples of VN→CN messages,

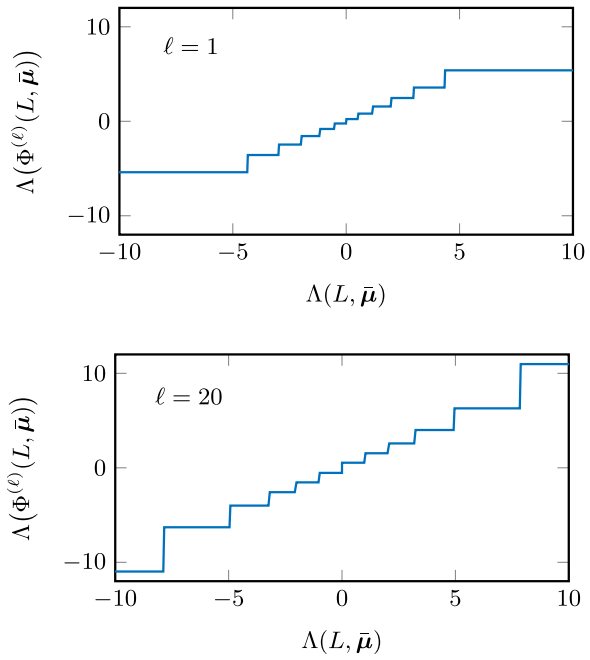

Fig. 2. LLR quantization induced by information-optimal symmetric VN LUTs for decoding iterations $\ell = 1$ (top) and $\ell = 20$ (bottom) for a (3, 6) LDPC code over a binary input AWGN channel with $\sigma = .84$. In spite of fixed message alphabets of size $|\mathcal{M}_\ell| = |\overline{\mathcal{M}_\ell}| = 12$ the associated LLR magnitudes increase during iterations.

CN→VN messages, and channel outputs (e.g., $\mathcal{Y} = \mathcal{L} \times \overline{\mathcal{M}_\ell}^{i-1}$ when designing the update $\Phi_i^{(\ell)}$ for VNs of degree i in iteration ℓ). We assume that the elements of $\mathcal{Y}$ are sorted according to increasing LLR, cf. (11), (14), and (15). Our goal of finding an information-optimal LUT is equivalent to finding a quantized discrete message update $\Phi : \mathcal{Y} \to \mathcal{Z}$ with $\mathcal{Z} = \{z_1, \ldots, z_K\}$ and $K \leq M$ that maximizes the mutual information (MI) [35] between the output message $\mathsf{z} = \Phi(\mathsf{y})$ and the relevance variable x,

$$\Phi^\star = \arg\max_{\Phi} I(\Phi(\mathsf{y}); \mathsf{x}). \tag{35}$$

This design has to be performed for all VN updates $\Phi_i^{(\ell)}$, $i \in \mathcal{D}_v$, and all CN updates $\bar{\Phi}_j^{(\ell)}$, $j \in \mathcal{D}_c$, for all iterations $\ell \in \{1, \ldots, \mathcal{L}\}$.

The convexity of the objective function in the maximization problem (35) and the separating hyperplane condition from [28] imply that the optimal quantized update $\Phi^\star$ is deterministic with contiguous quantization regions,

$$\Phi^\star(y_m) = z_k, \quad a_k \leq m < a_{k+1},$$

with quantization boundaries

$$0 = a_0 < a_1 < \cdots < a_{K-1} < a_K = M.$$

Finding the optimal quantizer then amounts to solving a dynamic program with worst-case complexity $O(M^3)$ [17], [28] (more efficient algorithms have recently been proposed in [36]–[38]). The quantizer inputs have symmetric distribution and even cardinality. To maintain these properties for the quantizer output, we restrict ourselves to symmetric quantizers,

$$\Phi(\neg y) = \neg\Phi(y). \tag{36}$$

Note that there may be non-symmetric quantizers that achieve a larger mutual information than the best symmetric quantizer at the cost of increased complexity. The restriction to symmetric quantizers ensures that we obtain a symmetric MP scheme (cf. (17), (18), (19)). Indeed, for the VN updates we have $\mathcal{Y} = \mathcal{L} \times \overline{\mathcal{M}}^{i-1}$ and according to (27) the inverse element of $y = (L, \overline{\boldsymbol{\mu}})$ is $\neg y = (\neg L, \neg\overline{\boldsymbol{\mu}})$. The relation (19) then follows directly from (36). For the CN LUTs, we have $\mathcal{Y} = \mathcal{M}^{j-1}$ and define the "non-negative" label set $\mathcal{M}_+^{j-1} = \{\boldsymbol{\mu} \in \mathcal{M}^{j-1} : \boldsymbol{\mu} = \mathrm{abs}(\boldsymbol{\mu})\}$ ($\mathrm{abs}(\cdot)$ is to be understood element-wise). Further, consider the partition

$$\mathcal{M}^{j-1} = \bigcup_{\boldsymbol{\mu} \in \mathcal{M}_+^{j-1}} (\mathcal{M}_{\boldsymbol{\mu}} \cup \neg\mathcal{M}_{\boldsymbol{\mu}}).$$

Here, $\mathcal{M}_{\boldsymbol{\mu}} = \{\boldsymbol{\mu}' : \mathrm{abs}(\boldsymbol{\mu}') = \boldsymbol{\mu},\ \mathrm{par}\,\boldsymbol{\mu}' = 1\}$ denotes the set of all tuples with (element-wise) magnitude $\boldsymbol{\mu}$ and even parity and $\neg\mathcal{M}_{\boldsymbol{\mu}} = \{\boldsymbol{\mu}' : \mathrm{abs}(\boldsymbol{\mu}') = \boldsymbol{\mu},\ \mathrm{par}\,\boldsymbol{\mu}' = -1\}$ is the corresponding set of odd-parity vectors. Using (33) together with $\tanh(\Lambda) = \mathrm{sgn}(\Lambda)\tanh(|\Lambda|)$, $\mathrm{sgn}(\Lambda(\mu)) = \mathrm{sign}(\mu)$, and $\Lambda(\neg\mu) = -\Lambda(\mu)$ implies that all tuples within $\mathcal{M}_{\boldsymbol{\mu}}$ have identical LLR and can thus be merged into a single LUT input y and similarly all elements of $\neg\mathcal{M}_{\boldsymbol{\mu}}$ can be merged into an inverse LUT input $\neg y$. Since any $\boldsymbol{\mu}' \in \neg\mathcal{M}_{\boldsymbol{\mu}}$ is related to an element $\tilde{\boldsymbol{\mu}} \in \mathcal{M}_{\boldsymbol{\mu}}$ via $\boldsymbol{\mu}' = \neg_{\boldsymbol{b}}\tilde{\boldsymbol{\mu}}$ with $\mathrm{par}\,\boldsymbol{b} = -1$, it follows from (28) that $p(\boldsymbol{\mu}'|+1) = p(\tilde{\boldsymbol{\mu}}|-1)$ and hence $p(\mathcal{M}_{\boldsymbol{\mu}}|+1) = p(\neg\mathcal{M}_{\boldsymbol{\mu}}|-1)$. The relation (18) is thus guaranteed by applying symmetric quantizers (36) to the merged input labels $\mathcal{M}_{\boldsymbol{\mu}}$ and $\neg\mathcal{M}_{\boldsymbol{\mu}}$. In terms of implementation this corresponds to separating sign and magnitude of the input message tuple according to (21) and (22) prior to the CN LUT updates. For both VN and CN product pmfs, label inversion corresponds to LLR value sign inversion, cf. (13), (31) and (33) . We further assume $p_{\mathsf{y}|\mathsf{x}}(\neg y|x) = p_{\mathsf{y}|\mathsf{x}}(y|-x)$ to ensure symmetry of the product pmfs (25).

Using the pre-image $\mathcal{Y}(z) \subset \mathcal{Y}$ of $z \in \mathcal{Z}$, the quantizer output distributions read

$$p_{\mathsf{z}|\mathsf{x}}(z|x) = \sum_{y \in \mathcal{Y}(z)} p_{\mathsf{y}|\mathsf{x}}(y|x), \quad p_{\mathsf{z}}(z) = \sum_{y \in \mathcal{Y}(z)} p_{\mathsf{y}}(y).$$

The (conditional) label distributions $p_{\mathsf{y}|\mathsf{x}}(y|x)$ and $p_{\mathsf{y}}(y)$ in practice have to be determined via DE. Using the symmetry constraints and the partial mutual information [28]

$$i(z) \triangleq p_{\mathsf{z}|\mathsf{x}}(z|1)\log_2\frac{p_{\mathsf{z}|\mathsf{x}}(z|1)}{p_{\mathsf{z}}(z)} + p_{\mathsf{z}|\mathsf{x}}(\neg z|1)\log_2\frac{p_{\mathsf{z}|\mathsf{x}}(\neg z|1)}{p_{\mathsf{z}}(z)},$$

we can develop the mutual information as

$$\begin{aligned} I(\mathsf{x};\mathsf{z}) &\triangleq \sum_{x \in \{-1,1\}} p_{\mathsf{x}}(x) \sum_{z \in \mathcal{Z}} p_{\mathsf{z}|\mathsf{x}}(z|x)\log_2\frac{p_{\mathsf{z}|\mathsf{x}}(z|x)}{p_{\mathsf{z}}(z)} \\ &= \sum_{z \in \mathcal{Z}_+} i(z), \end{aligned}$$

where $\mathcal{Z}_+ = \{z \in \mathcal{Z} : \mathrm{sign}(z) = 1\}$. The symmetry constraints thus allow us to restrict the quantizer design to the domain $\mathcal{Z}_+$, thereby effectively halving the number of quantizer labels. Rather than applying the algorithm from [28] to the complete set of input labels, we run the algorithm only for the "positive" input and output labels to obtain a mapping

Φ^+. Since the worst-case complexity of the algorithm from [28] scales cubically with the size of the input alphabet, the complexity of the LUT design is reduced by a factor up to $2^3 = 8$. The full symmetric LUT including "negative" message labels is obtained and implemented via the symmetry $\Phi(\neg y) = \neg\Phi(y) = \neg z$,

$$z = \Phi(y) = \begin{cases} \Phi^+(y), & \operatorname{sign}(y) \geq 0, \\ \neg\Phi^+(\neg y), & \operatorname{sign}(y) < 0. \end{cases}$$

Note that Φ^+ has only half the complexity of a non-symmetric LUT. For CN LUTs the savings are even larger since the inputs are preprocessed by merging the VN→CN messages $\boldsymbol{\mu}$ into $\mathcal{M}_{\boldsymbol{\mu}}$ and $\neg\mathcal{M}_{\boldsymbol{\mu}}$.

B. Min-LUT Decoder

We next explain how the algebraic structure and symmetry introduced in Section III can be exploited to replace the CN LUTs with a simpler $\min$ update (while retaining the VN LUTs). Specifically, we use (14), (21) and (22) to adapt the MS update rule (5) for discrete input labels $\boldsymbol{\mu} \in \mathcal{M}^{j-1}$ as[2]

$$\Phi(\boldsymbol{\mu}) = \begin{cases} \min(\operatorname{abs}(\boldsymbol{\mu})), & \operatorname{par}\boldsymbol{\mu} = 1, \\ \neg\min(\operatorname{abs}(\boldsymbol{\mu})), & \operatorname{par}\boldsymbol{\mu} = -1. \end{cases} \tag{37}$$

Note that by definition the output label alphabet equals $\mathcal{M}$. The resulting hybrid min-LUT decoding algorithm was first considered in our previous work [1], [25] for regular LDPC codes. The min-update (37) is more efficient than a LUT, which is particularly advantageous for high-rate codes that have CN degrees substantially larger than the VN degrees (e.g., the 10GBaseT regular LDPC code [10] has $d_v = 6$ and $d_c = 32$). The performance loss of the min-LUT approach (relative to pure LUT decoding) will be investigated in Section VI.

C. LUT Trees

Since the size of a LUT grows exponentially with the degree of the associated node, its design and implementation complexity may be prohibitive in practice. As a remedy, we propose to construct the LUT by nesting multiple lower-dimensional LUTs, each of which is designed using the approach from Section IV-A with the required pmfs obtained by applying (23) and (24) recursively. As an example, consider a VN of degree $i = 6$ with 4-bit messages. A general LUT $\Phi(L, \bar{\mu}_1, \ldots, \bar{\mu}_5)$ for such a VN has size $(2^4)^6 = 16\,777\,216$. If we restrict to the nesting

$$\Phi(L, \bar{\mu}_1, \ldots, \bar{\mu}_5) = \Phi_0\big(\Phi_1(\bar{\mu}_1, \bar{\mu}_2, \bar{\mu}_3), \Phi_2(\bar{\mu}_4, \bar{\mu}_5), L\big), \tag{38}$$

we only need two LUTs of size $(2^4)^3 = 4\,096$ (Φ_0 and Φ_1) and one LUT of size $(2^4)^2 = 256$ (Φ_2), thereby in total achieving a complexity-reduction by a factor of almost 2 000.

Any LUT nesting can be identified with a directed tree whose leaf nodes correspond to the LUT inputs, whose root node constitutes the LUT output, and whose inner nodes represent intermediate LUTs. An incoming edge represents an input and an outgoing edge corresponds to an output.

[2]For an arbitrary label set $\mathcal{Y}$, the minimum $y^* = \min\mathcal{Y}$ is defined by $y^* \prec y$ for all $y \in \mathcal{Y}$.

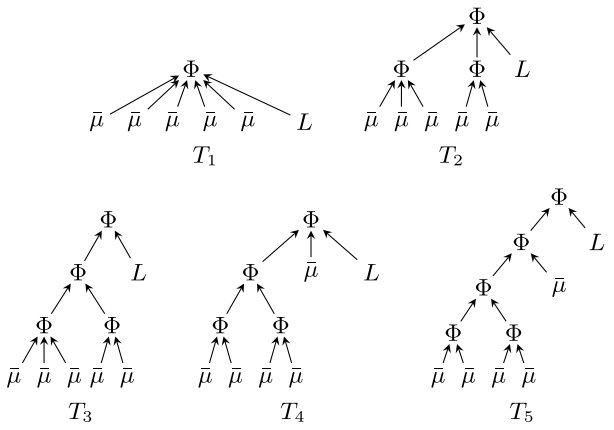

Fig. 3. Five different VN LUT tree structures. Note that $T_1 \succeq T_2 \succeq T_3$ (i.e., T_3 is a refinement of T_2, which in turn is a refinement of T_1) and $T_1 \succeq T_4 \succeq T_5$. However, T_2 and T_3 cannot be compared to T_4 and T_5.

For example, the general LUT $\Phi(L, \bar{\mu}_1, \ldots, \bar{\mu}_5)$ corresponds to tree T_1 in Figure 3 whereas the example nesting (38) corresponds to tree T_2. Since the input labels are iid, the order of the arguments in the nested LUT is immaterial. In view of the LLR quantization interpretation of LUTs, intermediate VN LUT updates can be thought of as a sum-and-quantize operation. Likewise, intermediate CN LUT updates correspond to a boxplus-and-quantize operation. Furthermore, the symmetry of the overall LUT can be ensured by designing symmetric intermediate LUTs. In what follows, we provide guidelines on how to choose the LUT tree structure based on information-theoretic arguments and a heuristic tree metric. The placement of the channel input L in the LUT tree is discussed at the end of this subsection.

1) Partial Tree Ordering: Let $\mathcal{Q}(T)$ denote the set of all LUTs that are realizable with the nesting induced by a tree T. We call a tree T_2 a refinement of T_1 (equivalently, T_1 a coarsening of T_2) iff $\mathcal{Q}(T_2) \subseteq \mathcal{Q}(T_1)$ (note that the input and output message alphabets are the same). Intuitively, a refined tree T_2 is obtained by placing new inner nodes between certain parent and child nodes in the tree T_1. Equivalently, a coarser tree T_1 is obtained by removing inner nodes in T_2 (neither leaves nor roots) and connecting the children of these nodes to the respective parent node. If a tree T_2 is a refinement of T_1, it admits for fewer realizable LUTs and hence

$$\max_{\Phi \in \mathcal{Q}(T_1)} I\big(\Phi(\mathsf{L}, \overline{\boldsymbol{\mu}}); \mathsf{x}\big) \geq \max_{\Phi \in \mathcal{Q}(T_2)} I\big(\Phi(\mathsf{L}, \overline{\boldsymbol{\mu}}); \mathsf{x}\big).$$

Thus, tree refinement defines a partial tree ordering $\succeq$ that induces a hierarchy in terms of maximum information flow. The ordering $\succeq$ is partial because there are pairs of trees where none is a refinement of the other.

2) Tree Depth: Since data processing can only reduce mutual information [35, Section 2.8], it is reasonable to require that the distance between leaf nodes (inputs) and root node (output) in a LUT tree should be small. We define the average depth δ of a LUT tree T as the arithmetic mean of these distances. DE simulations confirmed that the average depth is indeed a useful metric for ranking tree structures. Table II shows the average depth δ and the DE thresholds (for a (6, 32) regular LDPC code) obtained with the VN trees depicted

TABLE II

COMPARISON OF CUMULATIVE DEPTH AND DE THRESHOLD FOR VN LUT TREES FROM FIGURE 3 (ALL LUTS HAD 3 bit MESSAGE RESOLUTION AND THE MIN-LUT ALGORITHM WAS USED)

	T_1	T_2	T_3	T_4	T_5
δ	1	1.83	2.33	2.67	3.17
σ^*	0.5338	0.5328	0.5309	0.5313	0.5305

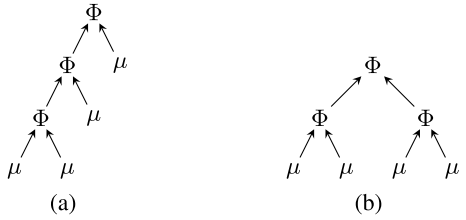

Fig. 4. Binary CN LUT trees with four inputs: (a) maximum-depth tree ($\delta = 2.25$); (b) minimum-depth tree ($\delta = 2$).

in Figure 3. Clearly, a larger δ implies a lower DE threshold (however, the differences in error rate performance are small).

Binary trees are practically attractive because the nested LUTs have only two inputs and hence minimal complexity. Previous work [16], [17], [21] was limited to *maximum-depth* binary trees (cf. Figure 4(a)) since the software implementation of such LUT trees only requires a single loop. While all binary LUT trees with m inputs require $m-1$ nested LUTs, there are two reasons why we advocate *minimum-depth* binary trees (cf. Figure 4(b)). First, by definition these binary trees have the smallest average tree depth δ, which—according to our simulations—leads to higher DE thresholds and lower error rates. Second, minimum-depth trees allow for parallel implementation of the the intermediate LUTs and hence lead to faster decoders [26]. Thus, all numerical simulations in this paper use minimum-depth binary LUT trees.

3) Placement of Channel LLR: During the initial decoder iterations, the channel LLRs carry much more information about the code bits than the CN→VN messages. Hence, for small ℓ, the channel LLR should be placed close to the root node in the binary LUT tree. As the decoding progresses, the CN→VN messages become more informative and hence for larger ℓ the channel LLR should be moved to the bottom of the LUT tree. Our simulations showed that this strategy indeed provides the best FER; however, if the total number of iterations is small ($\mathscr{L} \leq 20$), we observed that the loss induced by keeping the channel LLR next to the root of the LUT tree is negligible.

V. LUTS FOR IRREGULAR LDPC CODES

We next analyze and design LUT decoders for irregular LDPC codes. Here, distinct LUTs $\Phi_i^{(\ell)} : \mathcal{L} \times \overline{\mathcal{M}}_\ell^{i-1} \to \mathcal{M}_\ell$ and $\bar{\Phi}_j^{(\ell)} : \mathcal{M}_\ell^{j-1} \to \overline{\mathcal{M}}_{\ell+1}$ need to be designed for all VN degrees $i \in \mathcal{D}_v$, CN degrees $j \in \mathcal{D}_c$, and iterations $\ell = 1, \ldots, \mathscr{L}$. We incorporate the code's degree distributions $\lambda(\xi)$ and $\rho(\xi)$ (see (1)) into the information-optimal LUT design and rephrase the problem as an instance of the symmetric quantizer optimization problem discussed in Section IV-A. Based on the observation that degree distributions designed for conventional BP decoding [5], [39] are not well suited for LUT decoding, we further show how to optimize degree distributions for LUT decoding.

A. Joint LUT Design

To simplify the discussion, we do not distinguish between VN and CN updates and use a generic notation with node update Φ, degree index $d \in \mathcal{D}$, and degree distribution $p_{\mathsf{d}}(d)$.

Consider a LUT/quantizer $\Phi(\mathsf{y})$ whose input $\mathcal{Y}$ consists of the pairs (y_d, d), with y_d denoting the quantizer input label for a node with degree d. With the definition $\Phi_d(y_d) \triangleq \Phi(y_d, d)$, the quantizer/LUT $\Phi : \mathcal{Y} \to \mathcal{Z}$ can be interpreted as a collection of $|\mathcal{D}|$ individual LUTs/quantizers $\{\Phi_d\}_{d\in\mathcal{D}}$, where $\Phi_d : \mathcal{Y}_d \to \mathcal{Z}$ is the LUT for nodes of degree d. Rather than optimizing the quantizers $\{\Phi_d\}_{d\in\mathcal{D}}$ individually, we propose a joint design that maximizes the mutual information between code bit x and LUT (quantizer) output $\mathsf{z} = \Phi(\mathsf{y}_\mathsf{d}, \mathsf{d})$,

$$\Phi^* = \arg\max_{\Phi:\mathcal{Y}\to\mathcal{Z}} I(\Phi(\mathsf{y}), \mathsf{x}), \tag{39}$$

This has the advantage of taking into account the degree distribution $p_{\mathsf{d}}(d)$. More specifically, the distribution $p_{\mathsf{z},\mathsf{x}}(z,x)$ required to compute $I(\mathsf{z},\mathsf{x})$ is given by

$$\begin{aligned} p_{\mathsf{z},\mathsf{x}}(z,x) &= p_{\mathsf{x}}(x)p_{\mathsf{z}|\mathsf{x}}(z|x) = p_{\mathsf{x}}(x)\sum_{d\in\mathcal{D}} p_{\mathsf{z}|\mathsf{x},\mathsf{d}}(z|x,d)p_{\mathsf{d}}(d) \\ &= p_{\mathsf{x}}(x)\sum_{d\in\mathcal{D}}\sum_{y_d\in\Phi_d^{-1}(z)} p_{\mathsf{y}_d|\mathsf{x},\mathsf{d}}(y_d|x,d)p_{\mathsf{d}}(d). \end{aligned} \tag{40}$$

Here, $p_{\mathsf{y}_d|\mathsf{x},\mathsf{d}}(y_d|x,d)$ denotes the conditional label distribution for different degrees d and $\Phi_d^{-1}(z)$ is the pre-image of $z = \Phi_d(y_d)$. The conditional message distributions $p_{\mathsf{y}_d|\mathsf{x},\mathsf{d}}(y_d|x,d)$ again are determined via DE. The expression (40) explicitly involves an average with respect to the degree distribution $p_{\mathsf{d}}(d)$. The joint design (39) is superior to the following individual LUT designs

$$\max_{\Phi_d:\mathcal{Y}_d\to\mathcal{Z}} I(\Phi_d(\mathsf{y}_d), \mathsf{x}), \quad d \in \mathcal{D}, \tag{41}$$

since the latter does not take into account the degree distribution.

Applying the symmetric quantizer design algorithm from Section IV to (39) yields a symmetric quantizer $\Phi^\star$ with contiguous quantization regions on the LLRs of $(y_d, d) \in \mathcal{Y}$. Since

$$\begin{aligned} \Lambda(y_d, d) &= \log\frac{p_{\mathsf{y}_d,\mathsf{d}|\mathsf{x}}(y_d, d|+1)}{p_{\mathsf{y}_d,\mathsf{d}|\mathsf{x}}(y_d, d|-1)} \\ &= \log\frac{p_{\mathsf{y}_d|\mathsf{x},\mathsf{d}}(y_d|+1, d)}{p_{\mathsf{y}_d|\mathsf{x},\mathsf{d}}(y_d|-1, d)} = \Lambda(y_d) \end{aligned} \tag{42}$$

(due to the independence of the node degree d and code bit x), $\Phi^\star$ can be decomposed into $|\mathcal{D}|$ quantizers $\{\Phi_d^\star\}_{d\in\mathcal{D}}$, each of which has contiguous quantization regions and is symmetric. The quantizer input LLRs in (42) depend on the conditional message distribution for a given degree $p_{\mathsf{y}_d|\mathsf{x},\mathsf{d}}(y_d|x,d)$ but not on the degree distribution $p_{\mathsf{d}}(d)$. However, the probability of the joint event (y_d, d) depends on the degree distribution

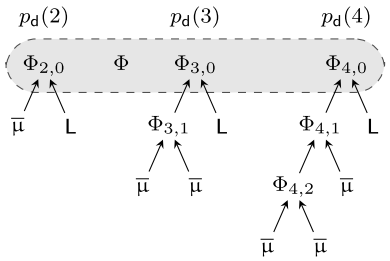

Fig. 5. Joint VN LUT tree design for an irregular LDPC code with VN degree distribution $p_{\mathsf{d}}(d)$, $d = 2, 3, 4$.

and thus different LUTs are optimal for different degree distributions. Specific attention must be paid to the design of LUT trees, as illustrated in the following example.

Example: Consider a code with VN degree distribution $p_{\mathsf{d}}(d)$, $d \in \{2, 3, 4\}$, and LUT tree structures shown in Figure 5. The intermediate LUTs $\Phi_{3,1}$, $\Phi_{4,1}$, and $\Phi_{4,2}$ are designed independently using the CN→VN message pmf $p_{\overline{\mu}|\mathsf{x}}$. However, the root node LUTs $\Phi_{d,0}$, $d = 2, 3, 4$, are designed jointly by taking into account the degree distribution and the pmf of the intermediate message z_d (the output of the quantizer $\Phi_{d,1}$). More specifically, we design the LUT $\Phi = \{\Phi_{2,0}, \Phi_{3,0}, \Phi_{4,0}\}$ according to (39) using the input triple (L, z_d, d) with joint distribution

$$p_{\mathsf{L},\mathsf{z},\mathsf{d}|\mathsf{x}}(L, z_d, d|x) = p_{\mathsf{z}|\mathsf{d},\mathsf{x}}(z_d|d, x) p_{\mathsf{L}|\mathsf{x}}(L|x) p_{\mathsf{d}}(d).$$

The performance difference between the joint design (39) and the individual design (41) can be significant. Figure 6 shows the binary input additive white Gaussian noise (BI-AWGN) DE thresholds [29] versus message bit resolution for two irregular LPDC codes with degree distributions specified in Table III when the decoder uses LUT trees designed with (39) ('joint') or (41) ('individual'). Clearly, the DE thresholds of both codes are larger when using the joint LUT design rather than the individual design, specifically with larger message resolution.

B. Degree Distributions for LUT Decoding

The code $\mathcal{C}_{\mathrm{BP}}$ in Table III and Figure 6 was taken from [5] and has been designed for BP decoding. Even with the joint design and at higher message resolutions, the DE threshold under LUT decoding for this code falls substantially short of the BP threshold. This suggests that this code ensemble is ill-suited for LUT decoding. Similar observations have inspired the design of irregular LDPC codes that mitigate the effects of quantization in MS decoding [40].

In what follows, we design LDPC codes with degree distributions optimized for LUT decoding. Special attention is paid to VNs with degree $i = 2$, which receive minimal extrinsic information (especially for quantized messages) and hence are most difficult to decode. For BP decoding, the asymptotic stability analysis in [5] provides an upper bound on the fraction of degree-2 edges. It seems intuitive that the fraction of degree-2 edges should be lower for LUT decoding, which can be viewed as a degraded form of BP. To verify this conjecture, we provide an extension of [5, Theorem 5] to LUT decoding

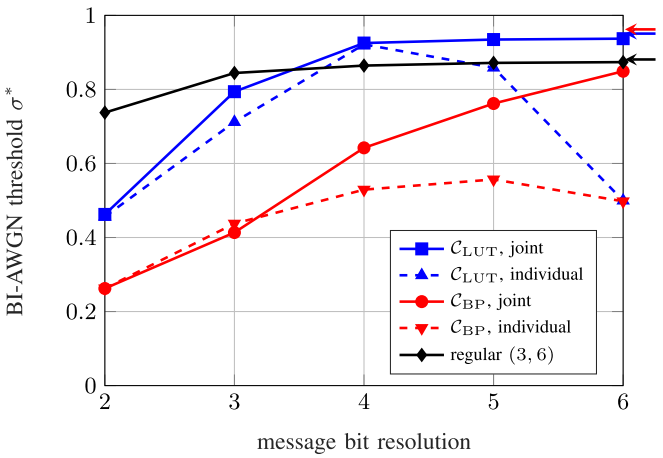

Fig. 6. DE thresholds versus message resolution for different LDPC ensembles in a BI-AWGN channel with individual and joint LUT design. Arrows on the right margin indicate the corresponding BP thresholds.

that states that the error probability is necessarily bounded away from 0 when the fraction of degree-2 edges is too large.

Lemma 2: Consider an LDPC code ensemble with degree distribution (λ, ρ) and symmetric channel pmf p_0. Let $p_\ell = \Phi_\ell(p_{\ell-1} \otimes p_0)$ for some sequence of quantizers Φ_ℓ, $\ell \in \mathbb{Z}$, and define

$$r \triangleq -\lim_{\ell \to \infty} \frac{1}{\ell} \log \mathrm{P_e}(p_\ell).$$

If $\lambda'(0)\rho'(1) > e^r$, then there exists a $\nu > 0$ such that $\mathrm{P_e} \geq \nu$ for any erasure-type message pmf.

Proof: The proof of this result can be found in [33]. It parallels that of [5, Theorem 5] and relies on Theorem 1 and on the duality between LUTs and LLR quantizers. ■

Unlike for BP decoding, Lemma 2 cannot be extended to arbitrary message pmfs due to the lack of optimality of LUT decoding. However, we found that the decoding bound $\lambda'(0) = \lambda_2 \leq \lambda_2^\star \triangleq e^r/\rho'(1)$ is still useful for searching LUT-optimized degree distributions. While for (unquantized) BP decoding there is the explicit expression $r = -\log \int_{\mathbb{R}} p_0(x) e^{-x/2} \, \mathrm{d}x$ [5], for LUT decoding we have to resort to numerical approximations. Figure 7 shows the convergence behavior of the decoding bound for both decoder types and the code ensemble $\mathcal{C}_{\mathrm{LUT}}$ from Table III. As expected, LUT decoding tolerates fewer degree-2 nodes. Furthermore, the decoding bound has converged after about 10^4 iterations.

C. Numerical Optimization of Degree Distribution

Our aim is to find degree distributions (λ, ρ) with maximum DE threshold under LUT decoding for a prescribed target rate R_t. We follow the hill climbing approach of [39], taking into account the decoding bound $\lambda_2^\star$ from the previous subsection (see [2] for details). We start with an initial degree distribution with rate $R < R_t$ and solve linear programs that maximize the code rate $R = 1 - \frac{\sum_j \frac{\rho_j}{j}}{\sum_i \frac{\lambda_i}{i}}$ by alternating minimization of $\sum_j \frac{\rho_j}{j}$ and maximization of $\sum_i \frac{\lambda_i}{i}$. Additional (linear) constraints limit the search space to the space of pmfs, restrict the step size to ensure linearity, and ensure that the degree

TABLE III

DEGREE DISTRIBUTIONS OF TWO RATE 1/2 CODES $\mathcal{C}_{\mathrm{BP}}$ (FROM [5]) AND $\mathcal{C}_{\mathrm{LUT}}$, OPTIMIZED FOR BP DECODING AND FOR MIN-LUT DECODING (SEE SECTION V-C), RESPECTIVELY. THE LAST TWO COLUMNS SHOW THE CORRESPONDING BI-AWGN DE THRESHOLDS FOR 4-bit MIN-LUT DECODING AND FOR FLOATING-POINT BP DECODING

	$\lambda(\xi)$	$\rho(\xi)$	$\sigma^{\star}_{\mathrm{LUT}}$	$\sigma^{\star}_{\mathrm{BP}}$
$\mathcal{C}_{\mathrm{BP}}$	$0.23802\,\xi + 0.20997\,\xi^2 + 0.03492\,\xi^3 + 0.12015\,\xi^4 + 0.01587\,\xi^6 + 0.00480\,\xi^{13} + 0.37627\,\xi^{14}$	$0.98013\,\xi^7 + 0.01987\,\xi^8$	0.6036	0.9622
$\mathcal{C}_{\mathrm{LUT}}$	$0.13805\,\xi + 0.40104\,\xi^2 + 0.02659\,\xi^8 + 0.43433\,\xi^{16}$	$0.32338\,\xi^7 + 0.67662\,\xi^8$	0.9292	0.9508

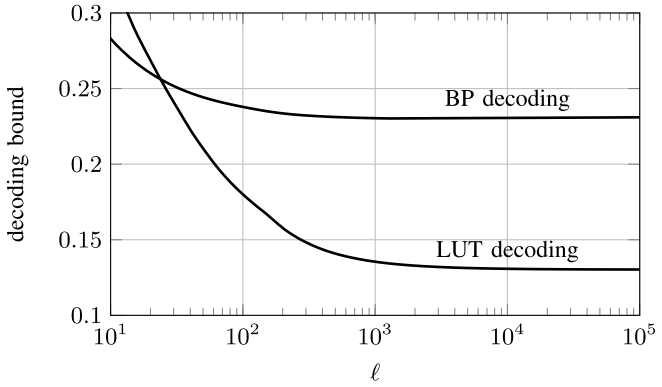

Fig. 7. Convergence behavior of decoding bound for BP and LUT decoding (with $\rho'(1) = 7.67662$, 4 bit messages, BI-AWGN channel with $\sigma = 0.929$).

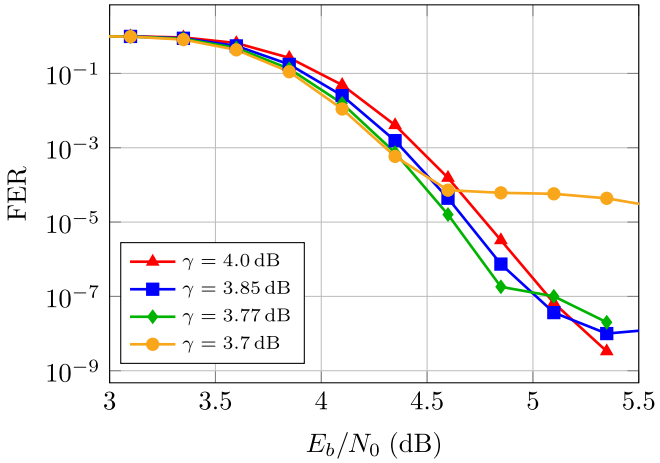

Fig. 8. FER versus E_b/N_0 for four different design SNRs γ (10GBASE-T Ethernet LDPC code with blocklength $N = 2048$ and rate $R = 0.84$ [10], min-LUT decoders with 3 bit channel LLRs, message alphabets downsized from 3 to 1 bit over $\mathscr{L} = 8$ decoding iterations, $\alpha = 50\%$ LUT reuse).

distribution update improves error probability [2], [39]. We further imposed the constraint $\lambda_2 \leq \lambda_2^*$, which was found to be necessary to achieve high DE thresholds.

VI. DESIGN ASPECTS OF LUT DECODERS

In this section, we discuss practical aspects of LUT decoder design like choosing the operating point, reusing LUTs over multiple iterations, and successive downsizing of the message alphabet.

A. Operating Point

The sequence $\Phi_i^{(\ell)}$, $\bar{\Phi}_j^{(\ell)}$ of information-optimal symmetric LUTs depends on the initial distribution $p_{\mathsf{L}|\mathsf{x}}(L|x)$ of the channel messages L and hence on the amount of channel noise.

For (quantized) BI-AWGN channels, our simulations indicated that a LUT decoder designed for noise level σ_0 works well at lower noise levels $\sigma < \sigma_0$, too. Indeed, the corresponding quantizer sequence is a bit too conservative but still achieves convergence (however, more iterations may be required). This observation is consistent with DE results and can be explained by the fact that LUTs act like quantized BP updates.

Let us define the BI-AWGN threshold

$$\sigma^*(\ell, P_t, \lambda, \rho) = \sup\{\sigma : P_e(\mathscr{L}) \leq P_t\}, \tag{43}$$

i.e., the highest noise level such that the error probability $P_e(\ell)$ after ℓ iterations is less than some target P_t (asymptotic thresholds such as those in Table III are obtained with $\ell \to \infty$ and $P_t \to 0$). For a practical LUT decoder design flow, P_t and a maximum number of iterations $\mathscr{L}$ are specified by the system requirements; The operating point (noise level) for the decoder is then chosen equal to the threshold $\sigma^*(\mathscr{L}, P_t, \lambda, \rho)$, which is computed via a DE bisection search. The operating point can equivalently be quantified by the corresponding design SNR $\gamma \triangleq -20\log_{10}(\sigma^*\sqrt{2R})$ [dB].

Figure 8 illustrates the tradeoff between error floor and waterfall performance obtained with various design SNRs. A lower design SNR corresponds to higher decoding threshold and hence better performance in the low SNR regime (waterfall region), whereas in the high SNR regime it is associated with higher error floors.

B. LUT Reuse

A difficulty with LUT decoders is the possibility that distinct LUTs are required in different iteration steps ℓ, i.e., in general $\Phi^{(\ell)} \neq \Phi^{(\ell')}$ for $\ell \neq \ell'$. This increases implementation complexity (as compared to BP and MS decoding) unless unrolled [25], [26] or serial [41] decoder architectures are used.

The problem with distinct LUTs is diminished by the fact that the number of necessary iterations with LUT decoding is actually smaller than with BP decoding. Figure 9 shows the BI-AWGN threshold (43) versus the number of iterations ℓ for the code ensembles from Table III and a regular (3,6) code under BP and LUT decoding. It is seen that the regular code has a smaller asymptotic threshold, which is achieved with around 40 iterations only. The irregular code ensemble $\mathcal{C}_{\mathrm{BP}}$

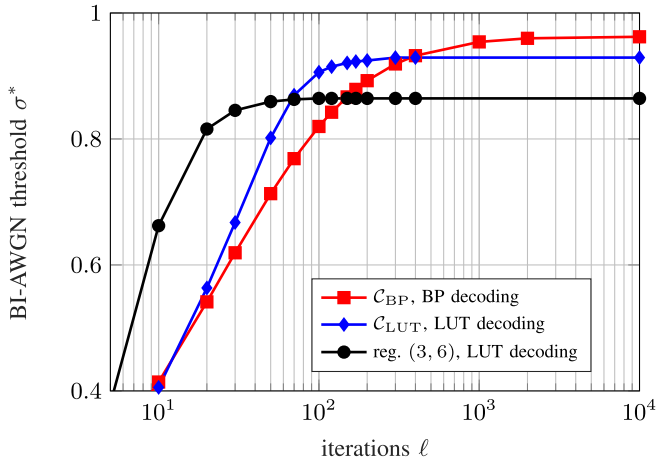

Fig. 9. BI-AWGN decoding threshold σ^* versus number of iterations ℓ for different code ensembles and decoders.

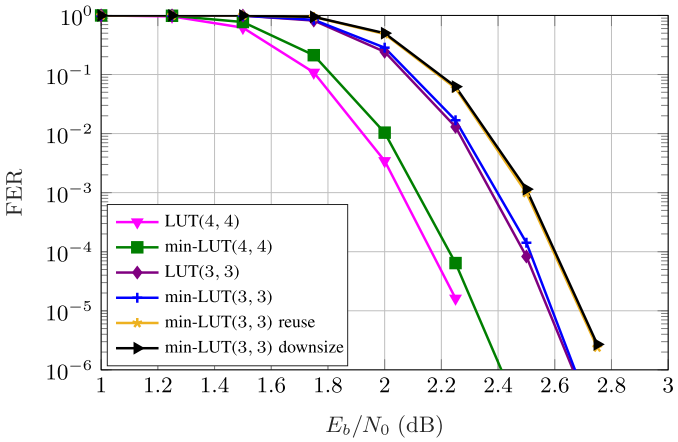

Fig. 10. FER versus E_b/N_0 achieved by various LUT and min-LUT decoders for a regular (3, 6) LDPC code of length $N = 5000$ with $\mathscr{L} = 20$ iterations; numbers in parenthesis indicate bit resolution for channel LLRs and message labels.

with floating-point BP decoding has the largest asymptotic threshold but requires about 1000 iterations. The asymptotic threshold of the irregular code ensemble $\mathcal{C}_{\mathrm{LUT}}$ is slightly smaller but already achieved with about 100 iterations; it is outperformed by the BP scheme only when more than 400 iterations are used.

To further reduce the number of distinct LUTs, we propose to reuse the LUTs designed for iteration ℓ in subsequent iterations $\ell' > \ell$. While this is not optimal, if the LUT reuse is designed appropriately, it can reduce decoder complexity substantially while only slightly deteriorating performance. We characterize the reuse pattern via a vector $\boldsymbol{r} \in \{0,1\}^{\mathscr{L}}$ such that $r_\ell = 1$ indicates that $\Phi^{(\ell)}$ is reused in iteration $\ell + 1$ and $r_\ell = 0$ implies a distinct LUT stage $\Phi^{(\ell+1)}$. The reuse pattern and the LUT design depend on each other and must be carefully designed. Simple heuristics such as reusing LUTs for a fixed number of iterations do not yield satisfactory results. We propose a greedy algorithm to find a reuse pattern with $\mathscr{L}_u$ distinct LUT stages (the reuse factor then equals $\alpha = 1 - \mathscr{L}_u/\mathscr{L}$): we start with $\boldsymbol{r} = \boldsymbol{0}$ and while $\boldsymbol{r}$ has more than $\mathscr{L}_u$ zeros we continue to set $r_\ell = 1$ for the iteration index ℓ for which a LUT reuse incurs the smallest performance penalty with regard to the current decoding threshold. The complexity of this algorithm scales cubically with $\mathscr{L}$ (more details and the actual algorithm can be found in [33, Section 4.4.2]).

C. Alphabet Downsizing

Another way to reduce implementation complexity is downsizing the message resolution, i.e., reducing the cardinality of the message alphabet over the decoder iterations, $|\mathcal{M}_{\ell+1}| < |\mathcal{M}_\ell|$. This concept is inspired by the fact that the distribution of the message labels becomes increasingly deterministic as the decoding iterations progress. As a consequence, smaller and smaller LUT quantizer resolutions can be used for these highly concentrated distributions without sacrificing much of decoder performance. Note that LUT reuse and alphabet downsizing can be combined, but a downsizing step and a LUT reuse cannot be performed in the same iteration. Error floors can be avoided even for short codes and combined LUT reuse and alphabet downsizing (cf. Figure 8).

VII. LUT Decoder Performance

We next compare different LUT decoder variants with conventional BP and MS decoders. In all figures, the labels LUT(b_1, b_2) and min-LUT(b_1, b_2) refer to decoders using b_1 bits for the channel LLR and b_2 bits for the messages. All LUT decoders were based on symmetric information-optimal minimum-depth binary LUT trees (see Subsection IV-C). Throughout this section, we use message resolution (number of bits per message) as a surrogate metric for hardware implementation complexity. The results shown in this section have been obtained using the UNFOLD software framework available at http://www.nt.tuwien.ac.at/UNFOLD.

A. LUT Decoder Variants

Figure 10 shows the frame error rate (FER) performance of several variants of (min-)LUT decoders (i.e., different message resolutions, LUT reuse, and alphabet downsizing) with $\mathscr{L} = 20$ iterations for a rate 1/2, regular (3, 6) LDPC code of length $N = 5000$ (designed using progressive edge growth). The min-LUT decoders perform only slightly worse than pure LUT decoders with the same resolution, with the gap vanishing as the message resolution decreases. Furthermore, reducing the message resolution from 4 bits to 3 bits deteriorates the FER performance for both decoder types by about 0.3 dB. The reuse-based decoder had reuse factor $\alpha = 0.8$ (i.e., only 4 distinct LUT stages) and performs about the same as the decoder that downsizes from 3-bit to 2-bit messages for the last 10 iterations. Both of these reduced-complexity decoders lose only about 0.1 dB compared to the LUT decoder with fixed 3-bit messages and no reuse. These FER results are in agreement with the DE threshold results (not shown). Indeed, we observed the SNR gaps between different decoders to match the gap between the corresponding DE decoding thresholds. Our FER simulations also confirmed that minimum-depth LUT trees are superior to maximum-depth trees.

TABLE IV

DEGREE DISTRIBUTION OF A RATE 1/2 DVB-S2-LIKE CODE $\bar{\mathcal{C}}_{\mathrm{BP}}$ AND A LUT-OPTIMIZED RATE 1/2 CODE $\bar{\mathcal{C}}_{\mathrm{LUT}}$ (SEE SECTION V-C). THE LAST TWO COLUMNS SHOW THE BI-AWGN DE THRESHOLDS FOR 4-bit MIN-LUT DECODING AND FOR FLOATING-POINT BP DECODING

	$\lambda(\xi)$	$\rho(\xi)$	$\sigma^{\star}_{\mathrm{LUT}}$	$\sigma^{\star}_{\mathrm{BP}}$
$\bar{\mathcal{C}}_{\mathrm{BP}}$	$0.30013\,\xi + 0.28395\,\xi^2 + 0.41592\,\xi^7$	$0.22919\,\xi^5 + 0.77081\,\xi^6$	0.5832	0.9497
$\bar{\mathcal{C}}_{\mathrm{LUT}}$	$0.16385\,\xi + 0.40637\,\xi^2 + 0.42978\,\xi^7$	$0.59105\,\xi^6 + 0.40876\,\xi^7 + 0.00019\,\xi^8$	0.8966	0.9178

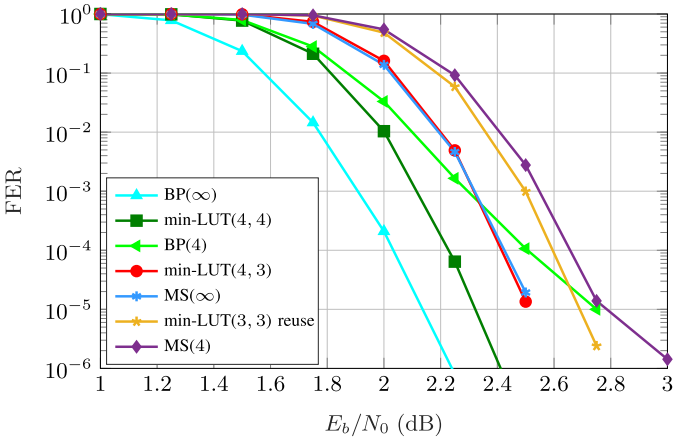

Fig. 11. FER versus E_b/N_0 achieved by various decoder architectures for a regular (3, 6) LDPC code of length $N = 5000$ with $\mathscr{L} = 20$ iterations; numbers in parenthesis indicate bit resolution ('∞' indicates floating point precision).

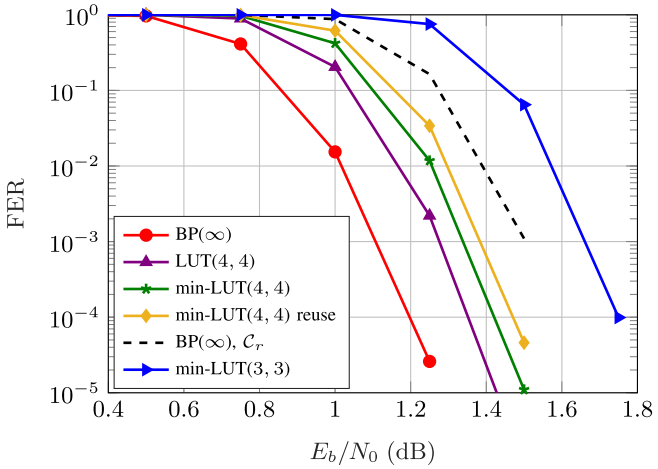

Fig. 12. FER versus E_b/N_0 achieved by various decoders for a rate 1/2 irregular code $\mathcal{C}_i$ of length $N = 10000$ (solid lines) and $\mathscr{L} = 100$ decoding iterations. For reference, we show BP decoding results of a rate 1/2 regular code $\mathcal{C}_r$ with the same length (dashed line).

B. LUT Versus MS/BP Decoders

We next compare min-LUT decoders to conventional BP and MS decoders (see Figure 11) for a regular (3, 6) LDPC code of length $N = 5000$ with $\mathscr{L} = 20$ iterations. The min-LUT decoder with 4-bit channel LLR and 4-bit message labels is only 0.2 dB worse than the BP decoder with floating-point precision and substantially better than the 4-bit fixed-point BP decoder. Decreasing the message resolution in the min-LUT decoder from 4 bit to 3 bit yields a performance on par with a conventional MS decoder at floating-point precision. Even with 3 bit channel LLRs and $\alpha = 80\%$ LUT reuse, the min-LUT decoder performs better than a fixed-point MS decoder using 4 bit messages.

Similar results were obtained for an irregular rate 1/2 LDPC code $\mathcal{C}_i$ of length $N = 10000$ using $\mathscr{L} = 100$ decoding iterations, cf. Figure 12. The code $\mathcal{C}_i$ was obtained via progressive edge growth from the $\mathcal{C}_{\mathrm{LUT}}$ ensemble in Table III. For reference, we also show a rate 1/2 regular code $\mathcal{C}_r$ of length $N = 10000$, designed in a similar manner from the regular (3, 6) ensemble, and decoded using 100 floating-point BP iterations. It is seen that the irregular code outperforms the regular code not only with BP decoding but also when decoded with 4 bit versions of the LUT decoder, the min-LUT decoder, and the min-LUT decoder with $\alpha = 80\%$ LUT reuse (the latter has a gap of 0.3 dB to floating-point BP). The performance advantage of $\mathcal{C}_i$ over $\mathcal{C}_r$ is lost only when the message resolution in the min-LUT decoder is decreased to 3 bit. This behaviour has also been observed in [40] and is consistent with our DE results, cf. Figure 6.

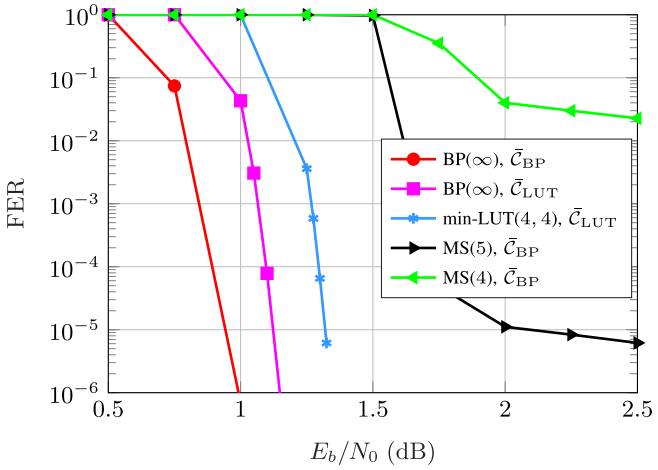

Fig. 13. FER Simulations for the LUT optimized code $\mathcal{C}_{\mathrm{LUT}}$ and the BP optimized DVB-S2 code $\mathcal{C}_{\mathrm{BP}}$. For for both codes $N = 64800$ and for all decoders $\mathscr{L} = 100$.

C. Application to DVB-S2

Our last simulation illustrates the potential of LUT-optimized codes and LUT decoders in a real-world DVB-S2-like scenario. More specifically, we compare a code $\bar{\mathcal{C}}_{\mathrm{BP}}$ whose degree distribution matches the DVB-S2 LDPC code [8] and a LUT-optimized code $\bar{\mathcal{C}}_{\mathrm{LUT}}$ (cf. Table IV). Both codes have rate 1/2 and length $N = 64800$. The FER results are shown in Figure 13. While $\bar{\mathcal{C}}_{\mathrm{BP}}$ with floating-point BP decoding performs best, using practically feasible 5-bit and

4-bit fixed-point MS decoders incurs an SNR penalty of about 1 dB and substantial error floors (higher resolutions offer negligible improvements as far as the waterfall region is concerned). These MS-decoded $\bar{\mathcal{C}}_{\mathrm{BP}}$ configurations are clearly outperformed (0.5 dB performance advantage) by the $\bar{\mathcal{C}}_{\mathrm{LUT}}$ ensemble with 4-bit min-LUT decoding, which offers excellent performance at very low complexity.

VIII. Conclusion

We presented a general framework for devising LUT (or, finite-alphabet, quantized BP) decoders for both regular and irregular LDPC codes. We developed a joint information-optimal symmetric LUT design to deal with distinct node degrees. To enable this design, we established a symmetry concept and an algebraic structure for discrete message labels. We further showed that a LUT decoder can be interpreted as a sequence of quantizers applied to BP updates. We proposed several means to further reduce LUT decoder complexity such as the use of hierarchical LUT trees, the min-LUT approximation, LUT reuse, and message alphabet downsizing.

Since codes with degree distributions optimized for BP decoding deteriorate with LUT decoding, we derived a stability bound for LUT decoding and we formulated an algorithm for designing irregular LDPC codes with degree distributions optimized for LUT decoding. The resulting codes also perform well under BP decoding and when the number of decoding iterations is limited.

Error rate simulations indicate that 4-bit min-LUT decoding of LUT-optimized irregular codes outperforms floating-point MS decoding of conventional codes. For regular LDPC codes, LUT decoders with message resolutions as low as 3 bit match the performance of floating-point MS decoders. The application of our framework for a full-fledged VLSI implementation of a 588 Gbps LDPC decoder for 10GBASE-T Ethernet is described in [26], confirming that our LUT decoders outperform MS decoding in terms of area and energy efficiency. Future work on similar hardware implementations of LUT decoders for other standards like DVB-S2 or 802.11n/ac is of great practical interest.

Appendix
Proof of Theorem 1

Let us consider the VN LUT update $\mu = \Phi_i(L, \bar{\boldsymbol{\mu}})$. Denote by $\mathcal{V}_\mu = \{(L, \bar{\boldsymbol{\mu}}) : \Phi_i(L, \bar{\boldsymbol{\mu}}) = \mu\} \subset \mathcal{L} \times \overline{\mathcal{M}}_{\mathrm{in}}^{i-1}$ the pre-image of μ, i.e., all input message tuples that are mapped to μ. The LLR value for the label μ is given by

$$\Lambda(\mu) = \Lambda\big(\Phi(L, \bar{\boldsymbol{\mu}})\big) = \log \frac{\sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)}{\sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)}.$$

Using the log-sum inequality [35] and the fact that a weighted average of a set of numbers is upper bounded by the largest element we have

$$\begin{aligned}\Lambda(\mu) &= \log \frac{\sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)}{\sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)} \\ &\le \sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)}{\sum_{(L',\bar{\boldsymbol{\mu}}')\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L', \bar{\boldsymbol{\mu}}'| + 1)} \\ &\qquad \times \log \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)}{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)} \\ &\le \max_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \log \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)}{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)} = \max_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \Lambda(L, \bar{\boldsymbol{\mu}}).\end{aligned} \tag{44}$$

$$\begin{aligned}&\qquad \times \log \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)}{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)} \\ &\le \max_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \log \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)}{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)} = \max_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \Lambda(L, \bar{\boldsymbol{\mu}}).\end{aligned} \tag{45}$$

Similarly,

$$\begin{aligned}-\Lambda(\mu) &= \log \frac{\sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)}{\sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)} \\ &\le \sum_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)}{\sum_{(L',\bar{\boldsymbol{\mu}}')\in\mathcal{V}_\mu} p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L', \bar{\boldsymbol{\mu}}'| - 1)} \\ &\qquad \times \log \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)}{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)} \\ &\le \max_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \log \frac{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| - 1)}{p_{\mathsf{L},\bar{\boldsymbol{\mu}}|\mathsf{x}}(L, \bar{\boldsymbol{\mu}}| + 1)} \\ &= \max_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} -\Lambda(L, \bar{\boldsymbol{\mu}}) = -\min_{(L,\bar{\boldsymbol{\mu}})\in\mathcal{V}_\mu} \Lambda(L, \bar{\boldsymbol{\mu}}),\end{aligned}$$

which together with (45) establishes (29). The proof of (30) is analogous, with $\Lambda(\mu)$ replaced by $\Lambda(\bar{\mu})$ and $\mathcal{V}_\mu$ replaced by $\mathcal{C}_{\bar{\mu}} = \{\boldsymbol{\mu} : \bar{\Phi}_j(\boldsymbol{\mu}) = \bar{\mu}\} \subset \mathcal{M}_{\mathrm{in}}^{j-1}$.

The identities in (31) follow trivially from (25) and (11). To prove (33), let $\boldsymbol{x} = (x_1, \ldots, x_{j-1}) \in \{-1, 1\}^{j-1}$ and define the index set $\mathcal{J}(\boldsymbol{x}, \boldsymbol{\mu}) = \{n : \operatorname{sign}(\mu_n) = x_n\}$. Without loss of generality we assume $p_{\mu|\mathsf{x}}(\mu|\operatorname{sign}(\mu)) \ge p_{\mu|\mathsf{x}}(\mu| - \operatorname{sign}(\mu))$. Then, (11) implies

$$|\Lambda(\mu)| = \log \frac{p_{\mu|\mathsf{x}}(\mu|\operatorname{sign}(\mu))}{p_{\mu|\mathsf{x}}(\mu| - \operatorname{sign}(\mu))}$$

and hence, with $\upsilon_n = \Lambda(\mu_n)$,

$$\begin{aligned}\prod_{n=1}^{j-1} p_{\Lambda(\mu)|\mathsf{x}}(\upsilon_n|x_n) &= \exp\left(\sum_{n\in\mathcal{J}(\boldsymbol{x},\boldsymbol{\mu})} |\upsilon_n|\right) \\ &\qquad \times \prod_{n=1}^{j-1} p_{\Lambda(\mu)|\mathsf{x}}(\upsilon_n| - \operatorname{sgn}(\upsilon_n)).\end{aligned}$$

Due to (12), we can further rewrite (26) as

$$p_{\boldsymbol{\mu}|\mathsf{x}}(\boldsymbol{\mu}|x) = C(\boldsymbol{\mu}) \sum_{\boldsymbol{x}:\,\mathrm{par}\,\boldsymbol{x}=x} \exp\left(\sum_{n\in\mathcal{J}(\boldsymbol{x},\boldsymbol{\mu})} |\Lambda(\mu_n)|\right)$$

where $C(\boldsymbol{\mu})$ is independent of x. Thus, the LLR of $\boldsymbol{\mu}$ can be computed as

$$\Lambda(\boldsymbol{\mu}) = \log \frac{\sum_{\boldsymbol{x}:\,\mathrm{par}\,\boldsymbol{x}=+1} \exp\left(\sum_{n\in\mathcal{J}(\boldsymbol{x},\boldsymbol{\mu})} |\upsilon_n|\right)}{\sum_{\boldsymbol{x}:\,\mathrm{par}\,\boldsymbol{x}=-1} \exp\left(\sum_{n\in\mathcal{J}(\boldsymbol{x},\boldsymbol{\mu})} |\upsilon_n|\right)}. \tag{46}$$

From (46), we see that the magnitude and the sign of $\Lambda(\boldsymbol{\mu})$ are determined, respectively, by the magnitude and the parity of $\boldsymbol{\upsilon} = (\upsilon_1, \ldots, \upsilon_{j-1})$. Thus, it is sufficient to consider the case where $\boldsymbol{\upsilon}$ has positive parity. For $k \in \{1, \ldots, j-1\}$, let us define

$$\Xi_k \triangleq \sum_{\mathcal{B}\in\mathcal{B}_k^{j-1}} \exp\left(\sum_{j\in\mathcal{B}} |\upsilon_j|\right),$$

where $\mathcal{B}_k^{j-1}$ denotes the set of all $\binom{j-1}{k}$ combinations of k distinct message labels. Using this definition, (46) becomes

$$\Lambda(\boldsymbol{\mu}) = \begin{cases} \log \dfrac{\Xi_{j-1} + \Xi_{j-3} + \cdots + \Xi_2 + 1}{\Xi_{j-2} + \Xi_{j-4} + \cdots + \Xi_1}, & j \text{ odd}, \\ \log \dfrac{\Xi_{j-1} + \Xi_{j-3} + \cdots + \Xi_2}{\Xi_{j-2} + \Xi_{j-4} + \cdots + \Xi_1 + 1}, & j \text{ even}. \end{cases}$$

We can combine these intermediate results into

$$\begin{aligned} \Lambda(\boldsymbol{\mu}) &= \log \frac{\prod_{n=1}^{j-1} \left(e^{|\upsilon_n|} + 1\right) + \prod_{n=1}^{j-1} \left(e^{|\upsilon_n|} - 1\right)}{\prod_{n=1}^{j-1} \left(e^{|\upsilon_n|} + 1\right) - \prod_{n=1}^{j-1} \left(e^{|\upsilon_n|} - 1\right)} \\ &= 2\operatorname{atanh}\left(\prod_{n=1}^{j-1} \tanh\left(\frac{\Lambda(\mu_n)}{2}\right)\right). \end{aligned}$$

Together with the sign inversion property of (46), this proves (33). ■

ACKNOWLEDGMENT

The authors are grateful to the reviewers and the Associate Editor, whose careful reading and numerous suggestions helped them to substantially improve the initial manuscript.

REFERENCES

[1] M. Meidlinger, A. Balatsoukas-Stimming, A. Burg, and G. Matz, "Quantized message passing for LDPC codes," in *Proc. 49th Asilomar Conf. Signals, Syst. Comput.*, Pacific Grove, CA, USA, Nov. 2015, pp. 1606–1610.

[2] M. Meidlinger and G. Matz, "On irregular LDPC codes with quantized message passing decoding," in *Proc. IEEE Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Sapporo, Japan, Jul. 2017, pp. 1–5.

[3] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[4] M. G. Luby *et al.*, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585–598, Feb. 2001.

[5] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[6] K. Zhao, W. Zhao, H. Sun, X. Zhang, N. Zheng, and T. Zhang, "LDPC-in-SSD: Making advanced error correction codes work effectively in solid state drives," in *Proc. 11th USENIX FAST*, 2013, pp. 243–256.

[7] J. Wang, T. Courtade, H. Shankar, and R. D. Wesel, "Soft information for LDPC decoding in flash: Mutual-information optimized quantization," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Houston, TX, USA, Dec. 2011, pp. 1–6.

[8] *Second Generation Framing Structure, Channel Coding and Modulation Systems for Broadcasting, Interactive Services, News Gathering and Other Broadband Satellite Applications*, document ETSI EN 302 307, Nov. 2014.

[9] *IEEE Standard for Air Interface for Broadband Wireless Access Systems*, Standard 802.16-2012, Aug. 2012.

[10] *IEEE Standard for Information Technology–Telecommunications and Information Exchange Between Systems–Local and Metropolitan Area Networks–Specific Requirements Part 3: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*, Standard 802.3as-2006, Sep. 2006.

[11] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, Aug. 2005.

[12] J. Zhao, F. Zarkeshvari, and A. H. Banihashemi, "On implementation of min-sum algorithm and its modifications for decoding low-density parity-check (LDPC) codes," *IEEE Trans. Commun.*, vol. 53, no. 4, pp. 549–554, Apr. 2005.

[13] T. Zhang, Z. Wang, and K. K. Parhi, "On finite precision implementation of low density parity check codes decoder," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Sydney, NSW, Australia, May 2001, pp. 202–205.

[14] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. J. Wainwright, "Design of LDPC decoders for improved low error rate performance: Quantization and algorithm choices," *IEEE Trans. Commun.*, vol. 57, no. 11, pp. 3258–3268, Nov. 2009.

[15] X. Zhang and P. H. Siegel, "Quantized min-sum decoders with low error floor for LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Cambridge, MA, USA, Jul. 2012, pp. 2871–2875.

[16] B. M. Kurkoski, K. Yamaguchi, and K. Kobayashi, "Noise thresholds for discrete LDPC decoding mappings," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, New Orleans, LO, USA, Nov./Dec. 2008, pp. 1–5.

[17] J. Lewandowsky and G. Bauch, "Trellis based node operations for LDPC decoders from the information bottleneck method," in *Proc. Int. Conf. Signal Process. Commun. Syst.*, Cairns, QLD, Australia, Dec. 2015, pp. 1–10.

[18] J. Lewandowsky, M. Stark, and G. Bauch, "Optimum message mapping LDPC decoders derived from the sum-product algorithm," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kuala Lumpur, Malaysia, May 2016, pp. 1–6.

[19] J. Lewandowsky and G. Bauch, "Information-optimum LDPC decoders based on the information bottleneck method," *IEEE Access*, vol. 6, pp. 4054–4071, Jan. 2018.

[20] D. Declercq, B. Vasic, S. K. Planjery, and E. Li, "Finite alphabet iterative decoders—Part II: Towards guaranteed error correction of LDPC codes via iterative decoder diversity," *IEEE Trans. Commun.*, vol. 61, no. 10, pp. 4046–4057, Oct. 2013.

[21] F. J. C. Romero and B. M. Kurkoski, "Decoding LDPC codes with mutual information-maximizing lookup tables," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 426–430.

[22] F. J. C. Romero and B. M. Kurkoski, "LDPC decoding mappings that maximize mutual information," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2391–2401, Sep. 2016.

[23] N. Slonim, N. Friedman, and N. Tishby, "Multivariate information bottleneck," *Neural Comput.*, vol. 18, no. 8, pp. 1739–1789, 2006.

[24] M. Stark, J. Lewandowsky, and G. Bauch, "Information-bottleneck decoding of high-rate irregular LDPC codes for optical communication using message alignment," *Appl. Sci.*, vol. 8, no. 10, p. 1884, 2018.

[25] A. Balatsoukas-Stimming, M. Meidlinger, R. Ghanaatian, G. Matz, and A. Burg, "A fully-unrolled LDPC decoder based on quantized message passing," in *Proc. IEEE Workshop Signal Process. Syst. (SiPS)*, Hangzhou, China, Oct. 2015, pp. 1–6.

[26] R. Ghanaatian, A. Balatsoukas-Stimming, T. C. Müller, M. Meidlinger, G. Matz, A. Teman, and A. Burg, "A 588-Gb/s LDPC decoder based on finite-alphabet message passing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 2, pp. 329–340, Feb. 2018.

[27] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[28] B. M. Kurkoski and H. Yagi, "Quantization of binary-input discrete memoryless channels," *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4544–4552, Aug. 2014.

[29] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[30] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[31] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 2, pp. 429–445, Mar. 1996.

[32] A. Winkelbauer and G. Matz, "On quantization of log-likelihood ratios for maximum mutual information," in *Proc. IEEE Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Stockholm, Sweden, Jun./Jul. 2015, pp. 316–320.

[33] M. Meidlinger, "Information-optimal decoding and demodulation on sparse graphs," Ph.D. dissertation, Technische Universität Wien, Vienna, Austria, 2018.

[34] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," in *Proc. 37th Allerton Conf. Commun., Control, Comput.*, Monticello, IL, USA, Sep. 1999, pp. 368–377.

[35] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Hoboken, NJ, USA: Wiley, 2006.

[36] K.-I. Iwata and S.-Y. Ozawa, "Quantizer design for outputs of binary-input discrete memoryless channels using SMAWK algorithm," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, Jun./Jul. 2014, pp. 191–195.
[37] H. Vangala, E. Viterbo, and Y. Hong, "Quantization of binary input DMC at optimal mutual information using constrained shortest path problem," in *Proc. 22nd Int. Conf. Telecommun.*, Sydney, NSW, Australia, Apr. 2015, pp. 151–155.
[38] X. He, K. Cai, W. Song, and Z. Mei, "Dynamic programming for sequential deterministic quantization of discrete memoryless channels," Jan. 2019, *arXiv:1901.01659*. [Online]. Available: https://arxiv.org/abs/1901.01659
[39] S.-Y. Chung, G. D. Forney, Jr., T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.
[40] B. Smith, F. R. Kschischang, and W. Yu, "Low-density parity-check codes for discretized min-sum decoding," in *Proc. Biennial Symp. Commun.*, Kigston, ON, Canada, May/Jun. 2006, pp. 14–17.
[41] E. Liao, E. Yeo, and B. Nikolic, "Low-density parity-check code constructions for hardware implementation," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, Jun. 2004, pp. 2573–2577.

Michael Meidlinger received the M.Sc. and Ph.D. degrees (Hons.) from Technische Universität (TU) Wien, Vienna, Austria, in 2013 and 2018 respectively.

From 2011 to 2013, he was working on the field of mobile communication research and contributed to developing the Vienna LTE-A Simulators. From 2013 to 2018, he was a part of the Communication Theory Group, TU Wien, where he did a research on quantizer design for telecommunication receivers as well as error correction coding and superposition modulation and coding techniques. Since 2018, he has been with Thales Austria, where he is working on safety critical cloud computing.

Gerald Matz (SM'07) received the Dipl.-Ing. and Dr. techn. degrees in electrical engineering and the Habilitation degree in communication systems from the Vienna University of Technology, Austria, in 1994, 2000, and 2004, respectively.

He currently holds a tenured Associate Professor position at the Institute of Telecommunications, Vienna University of Technology. He has held visiting positions with the Laboratoire des Signaux et Systèmes, Ecole Supérieure d'Electricité, France, in 2004, the Communication Theory Lab, ETH Zürich, Switzerland, in 2007, and Ecole Nationale Supérieure d'Electrotechnique, d'Electronique, d'Informatique et d'Hydraulique de Toulouse, France, in 2011. He has directed or actively participated in several research projects funded by the Austrian Science Fund (FWF), the Viennese Science and Technology Fund (WWTF), and the European Union. He has published some 220 scientific articles in international journals, conference proceedings, and edited books. He is the Co-Editor of the book *Wireless Communications over Rapidly Time-Varying Channels* (New York: Academic, 2011). His research interests include wireless networks, statistical signal processing, information theory, and big data.

Prof. Matz has served as a member of the IEEE SPS Technical Committee on Signal Processing Theory and Methods and the IEEE SPS Technical Committee on Signal Processing for Communications and Networking. He is a member of the ÖVE. He served as the General Chair of Asilomar 2019, the Technical Program Chair of Asilomar 2016, and the Technical Program Co-Chair of EUSIPCO 2004. He has been a member of the Technical Program Committee of numerous international conferences. In 2006, he received the Kardinal Innitzer Most Promising Young Investigator Award. He was an Associate Editor of IEEE SIGNAL PROCESSING LETTERS from 2004 to 2008, IEEE TRANSACTIONS ON SIGNAL PROCESSING from 2006 to 2010, *EURASIP Journal Signal Processing* from 2007 to 2010, and IEEE TRANSACTIONS ON INFORMATION THEORY from 2013 to 2015.

Andreas Burg (S'97–M'05) was born in Munich, Germany, in 1975. He received the Dipl.Ing. degree from the Swiss Federal Institute of Technology (ETH) Zürich, Zürich, Switzerland, in 2000, and the Dr. Sc. Techn. degree from the Integrated Systems Laboratory, ETH Zürich, in 2006.

In 1998, he worked at Siemens Semiconductors, San Jose, CA, USA. During his Ph.D. studies, he worked at Bell Labs Wireless Research for a total of one year. From 2006 to 2007, he was a Post-Doctoral Researcher with the Integrated Systems Laboratory and the Communication Theory Group, ETH Zürich. In 2007, he co-founded Celestrius, an ETH-spinoff in the field of MIMO wireless communication, where he was responsible for the ASIC development as the Director for VLSI. In January 2009, he joined ETH Zürich as a SNF Assistant Professor and the Head of the Signal Processing Circuits and Systems Group with the Integrated Systems Laboratory. In January 2011, he joined the Ecole Polytechnique Federale de Lausanne (EPFL), where he is leading the Telecommunications Circuits Laboratory. He was promoted to an Associate Professor with tenure in June 2018.

Mr. Burg is a member of the EURASIP SAT SPCN and the IEEE TC-DISPS and the CAS-VSATC. He has served on the TPC of various conferences on signal processing, communications, and VLSI. He was the TPC Co-Chair of VLSI-SoC 2012, ESSCIRC 2016, and SiPS 2017. He was the General Chair of ISLPED 2019. He served as an Editor for IEEE TRANSACTION OF CIRCUITS AND SYSTEMS in 2013 and on the Editorial Board of *Microelectronics Journal* (Springer). He is currently an Editor of *Journal on Signal Processing Systems*, (Springer), *MDPI Journal on Low Power Electronics and its Applications*, and IEEE TRANSACTIONS ON VLSI.