A Reconstruction-Computation-Quantization (RCQ) Approach to Node Operations in LDPC Decoding

Linfang Wang, Richard D. Wesel
University of California, Los Angeles
Department of Electrical and Computer Engineering
{lfwang,wesel}@ucla.edu

Maximilian Stark, Gerhard Bauch
Hamburg University of Technology
Institute of Communications
{maximilian.stark,bauch}@tuhh.de

***Abstract*—This paper proposes a finite-precision decoding method for low-density parity-check (LDPC) codes that features the three steps of Reconstruction, Computation, and Quantization (RCQ). Unlike Mutual-Information-Maximization Quantized Belief Propagation (MIM-QBP), RCQ can approximate either belief propagation or Min-Sum decoding. MIM-QBP decoders do not work well when the fraction of degree-2 variable nodes is large. However, sometimes a large fraction of degree-2 variable nodes is used to facilitate a fast encoding structure, as seen in the IEEE 802.11 standard and the DVB-S2 standard. In contrast to MIM-QBP, the proposed RCQ decoder may be applied to any off-the-shelf LDPC code, including those with a large fraction of degree-2 variable nodes. Simulations show that a 4-bit Min-Sum RCQ decoder delivers frame error rate (FER) performance within 0.1 dB of full-precision belief propagation (BP) for the IEEE 802.11 standard LDPC code in the low SNR region. The RCQ decoder actually outperforms full-precision BP and Min-Sum in the high SNR region were FER less than 10^{-5}. This paper also introduces Hierarchical Dynamic Quantization (HDQ) to design the non-uniform quantizers required by RCQ decoders. HDQ is a low-complexity design technique that is slightly sub-optimal. Simulation results comparing HDQ and optimal quantization on the symmetric binary-input memoryless additive white Gaussian noise channel show a mutual information loss of less than 10^{-6} bits, which is negligible for practical applications.**

***Index Terms*—Low Precision LDPC Decoder, Information Maximization Quantizer, Finite-Precision Decoding.**

I. Introduction

Low-Density Parity-Check (LDPC) codes [1], [2] have been widely used in wireless communication and NAND flash systems because of their excellent error correction capabilities. Message passing algorithms are used to decode LDPC codes, and in practical implementations the messages must often be quantized. Uniform quantization of messages with low precision significantly deteriorates decoder performance.

Recently, non-uniform quantization of messages in LDPC decoders have provided excellent performance with low precision and coarse quantization []. One way to realize non-uniform quantization LDPC decoders is to design lookup tables (LUTs) for variable nodes and/or check nodes. In [3], a Finite Alphabet Iterative Decoder (FAID) is proposed to overcome the error floor of an LDPC code under binary symmetric channel (BSC).

Aiming to minimize the performance degradation in the water fall region, [4] proposed a Mutual-Information-Maximization LUT (MIM-LUT) decoder. The MIM-LUT decomposes the actual node operation into a series of cascaded binary-input-single-output LUTs. In [5], Lewandowsky et al. proposed the Information-Optimum decoder, also called the Information Bottleneck (IB) decoder. Stark et al. extended the ideas from [4] and [5] to develop message alignment (MA) in [6]–[8] such that IB decoders can also work on irregular LDPC codes with arbitrary degree distributions. In [9], [10], the Min-LUT decoders were proposed, which replace the LUTs in the check node with a discrete, cluster-based Min-Sum operation. One problem faced by Min-LUT is it can't work well when the fraction of degree-2 variable node is large [10].

The other way to realize non-uniform quantization is by designing quantization parameters that maximize the mutual information between the source and quantized messages. In [11], Jason Kwok-San Lee and Jeremy Thorpe proposed a non-uniform BP decoder, which is implemented based only on simple mappings and fixed-point additions. Unfortunately, the authors did not provide a systematic way to find those mapping parameters. Recently, He et al. in [12] provided a systematic way to find mappings by implementing density evolution and dynamic programming quantization [13], and proposed MIM-QBP. They also extended MIM-QBP to irregular LDPC codes. However, similar to Min-LUT, MIM-QBP also faces the problem that it does not perform well when the fraction of degree-2 variable nodes in the LDPC code is large [12].

Even though both Min-LUT and MIM-QBP can have an excellent decoding performance by optimizing the edge distribution to lower the fraction of degree 2 variable nodes, it is sometimes necessary to consider LDPC codes with large fraction of degree 2 variable nodes. For an example, in the IEEE 802.11 standard's rate $1/2$ LDPC code, half of the variables nodes have degree 2 [14].

In this work, we generalize the structure in [11] and propose a finite-precision LDPC decoding method that features the three steps of Reconstruction, Computation, and Quantization (RCQ). Unlike MIM-QBP and Min-LUT, RCQ can be applied on any off-the-shelf LDPC code, including those with a large fraction of degree-2 variable nodes, such as the IEEE 802.11 code. The main contributions in this paper are:

This research is supported by National Science Foundation (NSF) grant CCF-1911166 Physical Optics Corporation (POC) and SA Photonics. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect views of the NSF, POC, or SA.

- We proposed a generalized RCQ decoder structure. Unlike the work in [11], [12], RCQ decoders can be an approximation of either BP decoders (*bp-RCQ*) or Min-Sum decoders (*ms-RCQ*).
- We designed an efficient sub-optimal quantization scheme, called Hierarchical Dynamic Quantization (HDQ), for the symmetric binary-input discrete memorelyess channel (BIDMC). HDQ is used for channel quantization and RCQ decoder construction.
- We used HDQ to implement Mutual Information Maximization Discrete Density Evolution (MIM-DDE), and showed that the RCQ decoder is a result of MIM-DDE.
- We designed a 4 bit *bp-RCQ* decoder for the IEEE 802.11 standard rate $1/2$ LDPC code for theoretical interests. Simulations show that a 4-bit *bp-RCQ* decoder delivers frame error rate (FER) performance less than 0.1dB of full-precision BP.
- We designed a 4 bit *ms-RCQ* decoder for the IEEE 802.11 standard rate $1/2$ LDPC code for practical implementation interests. Simulations show that a 4-bit *ms-RCQ* decoder delivers frame error rate (FER) performance 0.1dB of full-precision BP in the low SNR region. Simulation shows that *ms-RCQ* decoder actually outperforms full-precision BP and Min-Sum with FER less than or equal to 10^{-5}.

The remainder of this paper is organized as follows: In Sec. II, we give the description and notations for the RCQ decoder. A hierarchical dynamic quantization algorithm is proposed in Sec. III. Mutual information maximization Discrete Density Evolution is introduced in Sec. IV. This section also describes how to design RCQ decoders given an LDPC ensemble. Simulation results and discussion are given in Sec. V. Finally, Sec. VI concludes our work.

II. Reconstruction Computation Quantization Decoding Structure

Message passing algorithms update messages between variable nodes and check nodes in an iterative manner either until a valid codeword is found, or a predefined maximum number of iterations, I_T, is reached. The updating procedure contains two steps: 1) computation of the output , 2) message exchange of the output between neighboring nodes. We call messages with respect to the computation *internal message*, and messages passed over the edges of the Tanner graph *external message*. In [11], the authors proposed a LDPC decoder structure where the internal message has a higher precision than external message. In this work, we generalize their structure and propose a decoding framework that features three steps of Reconstruction, Computation and Quantization.

As illustrated in Fig.1, an RCQ decoder consists the following three parts:

1) Reconstruction: Reconstruction $R(\cdot) : \mathbb{F}_2^m \to \mathbb{F}_2^n (m < n)$ maps external message u_i to internal message r_i. We denote channel reconstruction by R^{ch}. Additionally, we denote variable node reconstruction and check node reconstruction at iteration t by R_t^c and R_t^v, respectively.

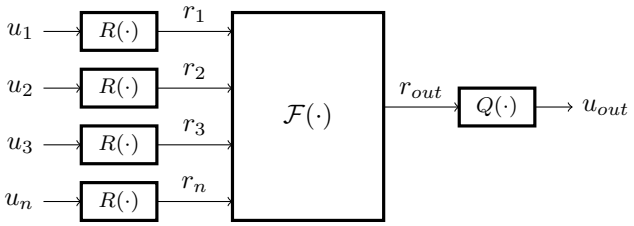

Fig. 1. RCQ Decoding Structure Illustration

2) Computaion: Computation function $\mathcal{F}(\cdot) : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is used to calculate the outgoing message r_{out}. We denote the variable node function and check node function by $\mathcal{F}^v$ and $\mathcal{F}^c$, respectively. $\mathcal{F}^v$ sums up all incoming messages. $\mathcal{F}^c$ can have different implementations. We denote the check node operation in BP (i.e. hyperbolic-tangent operation) and standard Min-Sum decoder by $\mathcal{F}_{bp}^c$ and $\mathcal{F}_{ms}^c$.

3) Quantization: A quantizer $Q(\cdot) : \mathbb{F}_2^n \to \mathbb{F}_2^m$ quantizes r_{out} to a m bit external message u_{out}. A m bit Quantizer Q is determined by $2^m - 1$ thresholds $\mathbf{th} = \{th_1, ..., th_{2^m-1}\}$ and

$$Q(i) = \begin{cases} 0 & i \leq th_1 \\ 2^m - 1 & i > th_{2^m-1} \\ j & th_j < i \leq th_{j+1} \end{cases} \quad (1)$$

We denote channel quantization by Q^{ch}, and check node quantization and variable node quantization at the i^{th} iteration by Q_i^c and Q_i^v respectively.

RCQ decoder precision can be fully described by a three tuple (m, n^c, n^v), which represents external message precision, check node internal message precision and variable node internal message precision. We use notation ∞ to denote floating point representation.

III. Hierarchical Dynamic Quantization

Like most non-uniform quantization LDPC decoders, designing RCQ decoder involves quantization that maximizes mutual information. Kurkoski in [13] proposed a dynamic programming method to find an optimal quantizer for BIDMC with complexity $\mathcal{O}(M^3)$, where M is cardinality of channel output. Dynamic programming quantization is proven to be optimal, however quantization becomes impractical when M is large. To mitigate computational complexity, different low-complexity near-optimal algorithms are proposed. In [15], Tal developed an annealing quantization algorithm with complexity $\mathcal{O}(M \log(M))$ for quantizing the symmetric BIDMC. In [5] Lewandowsky J. improved sequential Information Bottleneck algorithm (sIB) to quantize symmetric BIDMC . The computation complexity of the IB algorithm is $\mathcal{O}(tM)$, where t is the number of trials. As a machine learning algorithm, IB algorithm requires multiple trials to guarantee a satisfying result. In this work, we propose an efficient m bit quantization algorithm for the symmetric BIDMC with complexity $\mathcal{O}(mM)$.

Consider code bits $x \in \{0, 1\}$ in a binary LDPC codeword that are modulated by Binary Phase Shift Keying (BPSK), i.e.

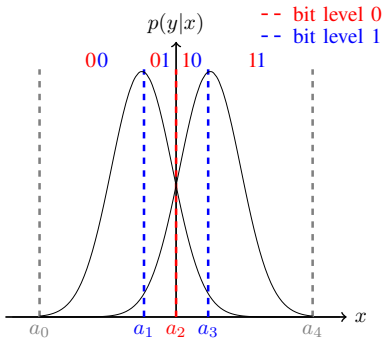

Fig. 2. HDQ method illustration: Quantizing symmetric BI-AWGNC observation into 2 bit messages

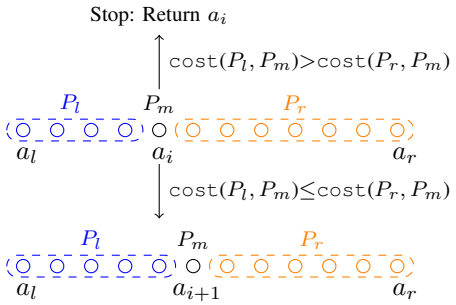

Fig. 3. An intermediate step of STS Algorithm

$s(x) = -2x + 1$, and transmitted over an Additive Gaussian White Noise (AWGN) channel. Assuming x obeys a uniform distribution and noise variance is σ^2, the joint probability density function between x and received signal y, $p(x, y|\sigma)$ is

$$p(x, y|\sigma) = \frac{1}{2\sqrt{2\pi\sigma^2}} e^{\frac{(y-s(x))^2}{2\sigma^2}}. \tag{2}$$

Since HDQ is designed under a BIDMC, we first uniformly quantize $p(x, y|\sigma)$ into M levels and denote the joint probability mass function (p.m.f.) by $P(X, Y), X = \{0, 1\}, Y = \{0, ..., M-1\}$. We denote $P(X = i, Y = i)$ by $P(X_i, Y_j)$ for simplicity.

A m bit Quantizer Q^{ch} aims to maximize the mutual information between X and quantized value T [13] :

$$\arg\max_{Q \in \mathcal{Q}} I(X; T). \tag{3}$$

Lemma 1 and Lemma 2 in [4] simplifies finding an optimal m bit quantizer to finding $2^m - 1$ boundaries $\{a_1, ..., a_{2^m-1}\}$. Even so, jointly optimizing $2^m - 1$ boundaries still has a large searching space. Hence, instead of optimizing thresholds jointly, the HDQ algorithm determines these boundaries *bit level by bit level*. Figure. 2 illustrates how HDQ quantizes symmetric BI-AWGNC output into 2 bit levels :

- initialize: a_0 and a_4.
- bit level 0: determine a_2, $a_0 < a_2 < a_4 - 1$,
- bit level 1 : fix a_2 and determine a_1 and a_3, $a_0 < a_1 < a_2 - 1$ and $a_2 < a_3 < a_4 - 1$.

Algorithm 1: Sequential Thresholds Searching (*STS*)

input : $P(X, Y)$, a_l, a_r
output: a_{out}
$P_l \leftarrow [P(X_0, Y_{a_l}) \quad P(X_1, Y_{a_l})]$
$P_m \leftarrow [P(X_0, Y_{a_l+1}) \quad P(X_1, Y_{a_l+1})]$
$P_r \leftarrow [\sum_{i=a_l+1}^{a_r-1} P(X_0, Y_i) \quad \sum_{i=a_l+1}^{a_r-1} P(X_1, Y_i)]$
for $i \leftarrow 1$ **to** $a_r - a_l - 2$ **do**
  $c_i^l \leftarrow \texttt{cost}(P_l, P_m)$
  $c_i^r \leftarrow \texttt{cost}(P_r, P_m)$
  if $c_i^l < c_i^r$ **then**
    $P_l \leftarrow P_l + P_m$
    $P_r \leftarrow P_r - P_m$
    $P_m \leftarrow [P(X_0, Y_{a_l+i+1}) \quad P(X_1, Y_{a_l+i+1})]$
  else
    return $a_l + i + 1$
  end
end
return $a_r - 1$

Algorithm 2: Hierarchical Dynamic Quantization

input : $\Pr(X, Y), X \in \{0, 1\}, Y \in \{0, ..., N-1\}$; m
output: $P(X, T)$, Q, R
$a_0 \leftarrow 0$
$a_N \leftarrow N - 1$
for $i \leftarrow 0$ **to** $m - 1$ **do**
  for $j \leftarrow 0$ **to** $2^{i-1} - 1$ **do**
    $a_{\frac{T}{2^i}(j+\frac{T}{2})} \leftarrow \texttt{STS}\left(a_{\frac{T}{2^i}j}, a_{\frac{T}{2^i}(j+1)}\right)$
  end
end
$P(X_i, T_j) \leftarrow \sum_{k=0}^{a_j-1} P(X_i, T_k)$
$th_i \leftarrow \log \frac{P(X_0, Y_{a_i})}{P(X_1, Y_{a_i})}$
$R(i) = \log \frac{P(X_0, T_i)}{P(X_1, T_i)}$

Note that a_1 and a_3 are independently optimized. It is easy to show that the solution of a_1 is independent to the solution of a_3. A similar idea is also used in optimizing progressive reads for flash memory cells [16].We borrow the metric of Information Bottleneck algorithm and develop a sequential threshold searching (STS) algorithm to find a_i. Given a_l and a_r, $r > l$ and starting from a_{l+1}, STS sequentially calculates the merging costs that each entry is merged into left or right cluster and stop when left merging cost is larger than right merging cost. Fig. 3 shows an intermediate step of STS. Merging cost is defined as mutual information loss when merging two probabilities together(Ref [5], Eq(10)) . Full description of STS and HDQ algorithm are given in Algorithm 1 and 2, respectively.

Fig. 4 shows various 4 bit quantization regions for channel output of BI-AWGNC under different σ^2. We examined four different quantization algorithms. Simulations show that the improved sIB algorithm and HDQ algorithm have a quantization result very close to the optimal dynamic programming

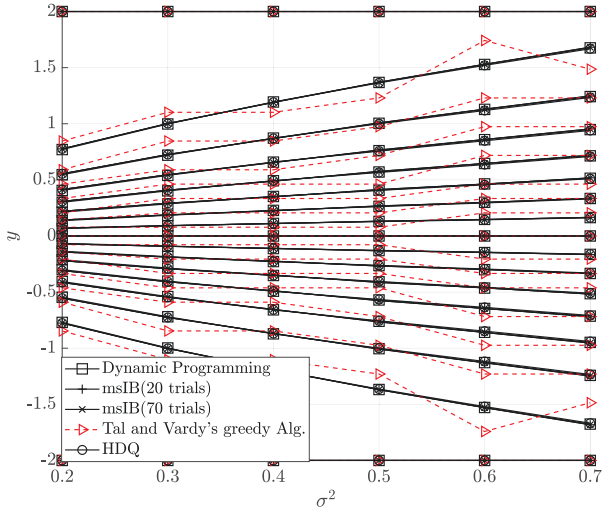

Fig. 4. Quantization regions for channel output of BI-AWGNC under different σ^2

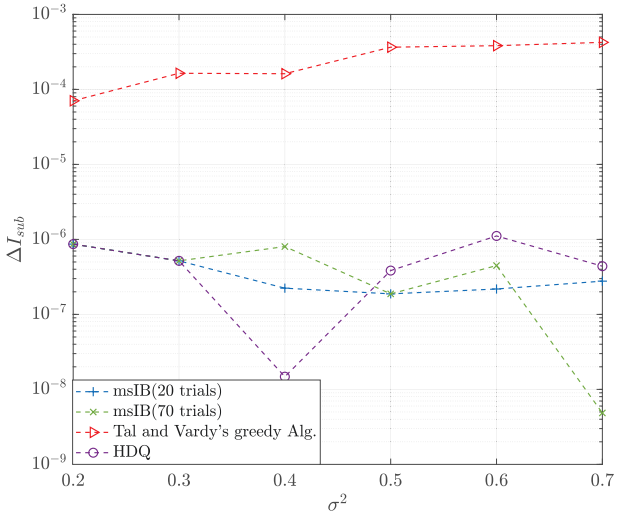

Fig. 5. Difference of mutual information loss between each sub-optimal quantizer and optimal quantizer

algorithm. Annealing quantization algorithm deviates from the optimal solution to different extent under different σ^2. We use $I^{dp}(X;T)$ to denote the mutual information between X and quantized value T, obtained by optimal dynamic programming quantizer and use $I^{sub}(X;T)$ to represent mutual information obtained through sub-optimal quantizers. Therefore, we can quantitatively evaluate the performance of each sub-optimal algorithm by $\Delta I_{sub} = I^{dp}(X;T) - I^{sub}(X;T)$.

Fig. 5 gives ΔI_{sub} of each sub-optimal quantizer. Simulations show that all three sub-optimal quantizers yield very similar mutual information to the optimal quantizer. However, we can still see that compared with annealing quantization, sIB algorithm and HDQ have a quantization result more close to the optimal quantizer because the ΔI_{sub} is around 10^{-6} for both sIB and HDQ.

In the next section, we will use HDQ to conduct mutual-information-maximization discrete density evolution and construct RCQ decoder.

IV. Mutual Information Maximization Discrete Density Evolution

RCQ decoder is a result of quantized density evolution : In the t^{th} iteration, Q_t^c, R_t^v, Q_t^v, R_{t+1}^v are constructed by quantizing the joint p.m.f. between code bits and the message from either the variable node or check node. To differentiate our discrete density evolution from the one using uniform quantization [17], we name our density evolution *Mutual-Information-Maximization Discrete Density Evolution* (MIM-DDE).

A. MIM-DDE at check node

Denote the joint p.m.f between the *external message* from the i^{th} variable node and corresponding code bit by $P^{v,i}(X,T)$, $X = \{0,1\}$, $T = \{0,...,2^m-1\}$. Based on the independence assumption in density evolution [18], all incoming messages have same distribution:

$$P^{v,i}(X,T) = P^{v}(X,T), \quad i = 0,...,d_c-2 \tag{4}$$

where d_c is check node degree. At check node, the code bit corresponding to output is the XOR sum of code bits corresponding to all inputs. By denoting:

$$P^{v,a}(X,T) \circledast P^{v,b}(X,T) \triangleq \sum_{\substack{m,n:\\ m \bigoplus n=k}} P^{v,a}(X_m,T)P^{v,b}(X_n,T), \tag{5}$$

where $m,n,k \in \{0,1\}$, the joint p.m.f between code bit corresponding to output and input messages, $P_{out}^c(X,\mathbf{T})$, can be represented by:

$$\begin{aligned} P_{out}^c(X,\mathbf{T}) &= P^{v,0}(X,T) \circledast ... \circledast P^{v,d_c-2}(X,T) && (6)\\ &= P^{v}(X,T) \circledast ... \circledast P^{v}(X,T) && (7)\\ &\triangleq P^{v}(X,T)^{\circledast(d_c-1)}, && (8) \end{aligned}$$

where $\mathbf{T}$ is a vector containing all incoming d_c-1 messages. Eq.(8) gives p.m.f. update when $\mathcal{F}_{bp}^c$ is implemented at the check node.

In order to keep the cardinality of external message the same, $P_{out}^c(X,\mathbf{T})$ needs to be quantized to 2^m levels. As pointed in [5], $|\mathbf{T}| = 2^{m(d_c-1)}$ will be very large when m and d_c is large. For an example, if $d_c = 8$ and $m = 4$, $|\mathbf{T}| = 2.68*10^8$. Hence, directly quantizing $P_{out}^c(X,\mathbf{T})$ is impossible. To mitigate the problem of *cardinality bombing*, we propose an intermediate coarse quantization algorithm called One-Step-Annealing (OSA) quantization without sacrificing mutual information. Note that Eq. (8) can be calculated in a recursive way and each step takes two inputs:

$$P_{out}^c(X,\mathbf{T})^{\circledast i} = P^{v}(X,T)^{\circledast(i-1)} \circledast P^{v}(X,T) \tag{9}$$

We observe that, in each step, the output of Eq.(9) has some entries with very close log likelihood ratio (LLR) values. By

Fig. 6. OSA illustration: points are ordered w.r.t. LLR values. Each color represents a cluster and LLR value difference in each cluster is less than l_s.

merging entries whose LLR difference is small enough, mutual information loss is negligible. Hence, OSA simply merges entries whose LLR values difference is less than a threshold l_s, and the output of OSA will be the input of the next p.m.f calculation step, i.e.:

$$P^v(X,T)^{\circledast i} = \texttt{OSA}(P^v(X,T)^{\circledast(i-1)}, l_s) \circledast P^v(X,T). \quad (10)$$

We use $l_s \in [10^{-4}, 10^{-3}]$ in our simulation. Fig. 6 shows an illustration of OSA and a full description of the OSA algorithm is given in Algorithm.3. The following table shows $|\mathbf{T}|$ after we implement OSA and choose different l_s. The example we show has the parameter $m = 4$, $d_c = 8$. The result shows that OSA greatly decreases the output cardinality, and based on our simulation, mutual information losses under these three l_s are all less than 10^{-7} bits.

l_s	0	10^{-4}	$5*10^{-4}$	10^{-3}
$\lvert\mathbf{T}\rvert$	$2.68*10^8$	$3.3*10^4$	$1.7*10^3$	$1.3*10^3$

For a regular LDPC code with check node degree d_c, HDQ is implemented to quantize $\mathbf{T}$ into a m bit message. We denote the joint p.m.f. between code bit x and quantized value T by $P^c(X,T)$. As a result of HDQ, Q^c and R^v in this iteration are constructed.

Unlike regular LDPC codes, irregular LDPC codes have different node types, we denote the check node edge distribution by $\rho(x) = \sum_{i=2}^{d_{c,max}} \rho_i x^{i-1}$. To update $P^c(X,T)$ and construct Q^c and R^v for irregular LDPC code, we need to quantize:

$$P^c_{out}(X,\mathbf{T}) = \sum_{i=2}^{d_c} \rho_i P^c(X,T)^{\circledast(i-1)} \quad (11)$$

Due to space limitations, we refer [9] to Min-Sum operation. Note that Min-Sum operation doesn't increase the cardinality of output, this implies for *ms-RCQ*:

1) $m = n^c$,
2) R^c doesn't change with respect to iteration. We can map 2^m messages to $\{-2^m-1, ..., -1, 1, ..., 2^m-1\}$ and then implement $\mathcal{F}^c_{ms}$. We can also implement a single LUT to realize the min-sum operation.

B. MIM-DDE at variable node

Each variable node sums the LLR messages from its channel observation and neighboring check nodes. By denoting:

$$P^{c,a}(X,T) \boxdot P^{c,b}(X,T) = \frac{1}{P(X)} P^{c,a}(X,T) P^{c,b}(X,T), \quad (12)$$

Algorithm 3: One Step Annealing Algorithm (OSA)

input : $\Pr(X,Y), X \in \{0,1\}, Y \in \{0,...,N-1\}$; l_s
output: $\Pr(X,T)$
$j \leftarrow 0$
$\Pr(X_0,T_j) \leftarrow P(X_0,Y_0)$
$\Pr(X_1,T_j) \leftarrow P(X_1,Y_0)$
$l \leftarrow \log \frac{\Pr(X_0,Y_0)}{\Pr(X_1,Y_0)}$
for $i \leftarrow 1$ **to** $N-1$ **do**
  if $(\log \frac{P(X_0,T_i)}{P(X_1,T_i)} - l) \le l_s$ **then**
    $P(X_0,T_j) \leftarrow \Pr(X_0,T_j) + \Pr(X_0,Y_i)$
    $P(X_1,T_j) \leftarrow \Pr(X_1,T_j) + \Pr(X_1,Y_i)$
  else
    $j \leftarrow j+1$
    $\Pr(X_0,T_j) \leftarrow \Pr(X_0,Y_i)$
    $\Pr(X_1,T_j) \leftarrow \Pr(X_1,Y_i)$
    $l \leftarrow \log \frac{\Pr(X_0,Y_i)}{\Pr(X_1,Y_i)}$
  end
end

the joint p.m.f between code bit X and incoming message combination $\mathbf{T}$, $P^v_{out}(X,\mathbf{T})$, given variable node degree d_v, can be expressed by:

$$P^v_{out}(X,\mathbf{T}) = P^{ch}(X,T) \boxdot P^c(X,T)^{\boxdot(d_v-1)}, \quad (13)$$

Similarly, for irregular LDPC codes with variable edge degree distribution $\lambda(x) = \sum_{i=2}^{d_{v,max}} \lambda_i x^{i-1}$, $P^v_{out}(X,\mathbf{T})$ is given by:

$$P^v_{out}(X,\mathbf{T}) = P^{ch}(X,T) \boxdot \sum_{i=2}^{d_{v,max}} \lambda_i P^c(X,T)^{\boxdot(d_v-1)}. \quad (14)$$

$P^v_{out}(X,\mathbf{T})$ is then quantized to 2^m levels by HDQ. Also, as a result of HDQ, and joint p.m.f between code bit X and quantized messages T, $P^v(X,T)$, is updated. Q^v in this iteration and R^c in the next iteration are built correspondingly. Note that variable nodes also face the *cardinality bombing* problem, hence OSA is needed in each recursive step.

Thus, by implementing MIM-DDE, we can iteratively update $P^c(X,T)$, $P^v(X,T)$ and build Q^c_i, Q^v_i, R^c_i and R^v_i, $i = \{0,...,I_T-1\}$.

In MIM-DDE, we only limit the precision of external messages, i.e. m, and keep internal messages, n^c (only for *bp-RCQ*) and n^v, full precision. To make internal message precision finite, a uniform n^c (or n^v) quantizer is required when implementing $\mathcal{F}^c$(or $\mathcal{F}^v$).

V. Simulation and Discussion

In this section, we build RCQ decoder for IEEE 802.11 standard LDPC code with codeword length 1296 and edge distribution:

$$\lambda(x) = 0.2588x + 0.3140x^2 + 0.0465x^3 + 0.3837x^{10}, \quad (15)$$
$$\rho(x) = 0.8140x^6 + 0.1860x^7. \quad (16)$$

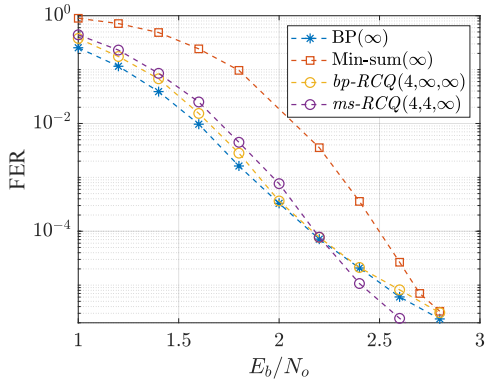

Fig. 7. RCQ decoder with full precision internal message

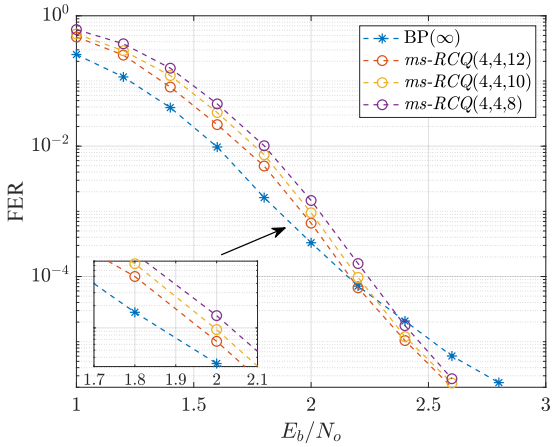

Fig. 8. The effect of internal message quantization for 4 bits *ms-RCQ*

This LDPC code has fast encoding structure hence half the variable nodes has degree 2. The $\frac{E_b}{N_o}$ we used to design RCQ is 0.90 dB for both *bp-RCQ* and *ms-RCQ*. I_T is set to be 50.

Fig. 7 shows the FER simulation result of *bp-RCQ*(4,∞,∞) and *ms-RCQ*(4,4,∞). As comparison, we give the performance of BP(∞) and Min-Sum (∞). BP decoder performs best in low $\frac{E_b}{N_o}$ region. However FER curve slope decreases after 2.4 dB due to elementary trapping sets [3], which is a result of many degree-2 variable nodes. Min-Sum(∞) decoder performs worst. For RCQ decoders, when $\frac{E_b}{N_o}$ is low, compared with BP(∞), *bp-RCQ* (4,∞,∞) and *ms-RCQ*(4,4,∞) has a degradation 0.1 dB. As $\frac{E_b}{N_o}$ increases, *bp-RCQ*(4,∞,∞) behaves similarly to BP(∞) and have a degradation less than 0.1 dB. However, *ms-RCQ* (4,4,∞) outperforms BP(∞) and Min-Sum(∞) with FER less than or equal to 10^{-5}.

It seems counter intuitive that *ms-RFQ* performs even better than Min-Sum (∞), however it is worthy noting that *ms-RFQ* implements cluster-based Min-Sum operation rather than LLR based Min-Sum operation [10]. We collected noised codewords that BP could not decode under 2.6 dB and fed them into *ms-RCQ*. Simulation results show that *ms-RCQ* can decode 80% of them. Hence *ms-RFQ* is able to overcome elementary trapping set that degrades BP performance.

For a purpose of practical implementation, we are more interested in *ms-RCQ*. Fig.8 gives FER performance of *ms-RCQ* decoder with different n^v. When $\frac{E_b}{N_o} < 2.2$ dB, *ms-RCQ*(4,4,12) (5 bits to integer part and 7 bits to fraction part), *ms-RCQ*(4,4,10) (5 bits to integer part and 5 bits to fraction part) and *ms-RCQ*(4,4,8) (5 bits to integer part and 3 bits to fraction part) have a degradation 0.1, 0.15 and 0.2 dB, compared with BP(∞). When $E_b/N_o > 2.4$ dB, all three *ms-RCQ* decoders outperform BP(∞).

VI. Conclusion

In this work, HDQ is proposed to quantize a symmetric binary input discrete channel into m bit levels. Then we use HDQ and MIM-DDE to construct the RCQ decoder. Unlike MIM-QBP, RCQ can approximate either belief propagation or Min-Sum decoding. We use an IEEE 802.11 standard LDPC code to illustrate that the RCQ decoder works well when the fraction of degree-2 variable nodes is large. Simulations show that a 4-bit *ms-RCQ* decoder delivers frame error rate (FER) performance 0.1 dB of full-precision belief propagation (BP) in the low SNR region. The RCQ decoder actually outperforms full-precision BP in the high SNR region because it overcomes elementary trapping sets that decrease the slope of the FER curve under BP decoding at high SNR.

References

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, January 1962.

[2] W. Ryan and S. Lin, *Channel Codes: Classical and Modern*. Cambridge University Press, Sep. 2009.

[3] S. K. Planjery, D. Declercq, L. Danjean, and B. Vasic, "Finite alphabet iterative decoders, part i: Decoding beyond belief propagation on BSC," Jul. 2012.

[4] 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.

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

[6] M. Stark, J. Lewandowsky, and G. Bauch, "Information-Optimum LDPC decoders with message alignment for irregular codes," in *2018 IEEE Global Communications Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.

[7] M. Stark, L. Wang, R. D. Wesel, and G. Bauch, "Information bottleneck decoding of Rate-Compatible 5G-LDPC codes," Jun. 2019.

[8] M. Stark, L. Wang, G. Bauch, and R. Wesel, "Decoding rate-compatible 5g-ldpc codes with coarse quantization using the information bottleneck method," *IEEE Open Journal of the Comm. Society*, pp. 1–1, 2020.

[9] M. Meidlinger, A. Balatsoukas-Stimming, A. Burg, and G. Matz, "Quantized message passing for LDPC codes," in *2015 49th Asilomar Conf. on Signals, Systems and Computers*, Nov. 2015, pp. 1606–1610.

[10] M. Meidlinger and G. Matz, "On irregular LDPC codes with quantized message passing decoding," in *2017 IEEE 18th International Workshop on Signal Proc. Adv. in Wireless Comm. (SPAWC)*, Jul. 2017, pp. 1–5.

[11] J. K. . Lee and J. Thorpe, "Memory-efficient decoding of LDPC codes," in *Proc. Int. Symp. on Info. Theory, ISIT 2005.*, Sep. 2005, pp. 459–463.

[12] X. He, K. Cai, and Z. Mei, "Mutual Information-Maximizing quantized belief propagation decoding of LDPC codes," Apr. 2019.

[13] 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.

[14] M. Mankar, G. Asutkar, and P. Dakhole, "Reduced complexity quasi-cyclic ldpc encoder for ieee 802.11 n," *International Journal of VLSI design & Communication Systems (VLSICS)*, vol. 7, no. 5/6, 2016.

[15] I. Tal and A. Vardy, "How to construct polar codes," May 2011.

[16] N. Wong, E. Liang, H. Wang, S. V. S. Ranganathan, and R. D. Wesel, "Decoding flash memory with progressive reads and independent vs. joint encoding of bits in a cell," in *2019 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2019, pp. 1–6.

[17] S.-Y. Chung, G. D. Forney, 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.

[18] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. on information*, 2001.