Generalized Low-Density Parity-Check Codes Based on Hadamard Constraints

Guosen Yue, *Member, IEEE*, Li Ping, *Member, IEEE*, and Xiaodong Wang, *Senior Member, IEEE*

***Abstract*—In this paper, we consider the design and analysis of generalized low-density parity-check (GLDPC) codes in AWGN channels. The GLDPC codes are specified by a bipartite Tanner graph, as with standard LDPC codes, but with the single parity-check constraints replaced by general coding constraints. In particular, we consider imposing Hadamard code constraints at the check nodes for a low-rate approach, termed LDPC-Hadamard codes. We introduce a low-complexity message-passing based iterative soft-input soft-output (SISO) decoding algorithm, which employs the *a posteriori* probability (APP) fast Hadamard transform (FHT) for decoding the Hadamard check codes at each decoding iteration. The achievable capacity with the GLDPC codes is then discussed. A modified LDPC-Hadamard code graph is also proposed. We then optimize the LDPC-Hadamard code ensemble using a low-complexity optimization method based on approximating the density evolution by a one-dimensional dynamic system represented by an extrinsic mutual information transfer (EXIT) chart. Simulation results show that the optimized LDPC-Hadamard codes offer better performance in the low-rate region than low-rate turbo-Hadamard codes, but also enjoy a fast convergence rate. A rate-0.003 LDPC-Hadamard code with large block length can achieve a bit-error-rate (BER) performance of 10^{-5} at -1.44 dB, which is only 0.15 dB away from the ultimate Shannon limit (-1.592 dB) and 0.24 dB better than the best performing low-rate turbo-Hadamard codes.**

***Index Terms*—Code optimization, extrinsic mutual information transfer (EXIT) chart, generalized low-density parity-check (GLDPC) codes, low-complexity decoding, low-density parity-check (LDPC)-Hadamard codes, low-rate.**

I. Introduction

SINCE the discovery of turbo codes [1], error-correction codes approaching the Shannon capacity have attracted increasing interest. One of the important milestones has been the rediscovery of low-density parity-check (LDPC) codes [2], which were originally proposed by Gallager [3]. Irregular LDPC codes were introduced in [4], and were shown to asymptotically achieve the capacity of a binary erasure channel under iterative message-passing decoding. The design and performance analysis of irregular LDPC codes were treated in [5], [6] for memoryless channels based on density evolution and Gaussian approximation. It has also been shown that carefully designed irregular LDPC codes achieve performance that is very close to the Shannon capacity on additive white Gaussian noise (AWGN) channels and many other types of channels [7].

A standard LDPC code is characterized by the random connection of variable nodes and check nodes. This principle can be generalized. As first described in [8], the long codes can be constructed from different shorter subcodes other than single parity check (SPC) codes using bipartite graphs. For example, the SPC constraints at the check nodes in the standard LDPC code can be replaced by constraints based on other block codes, such as Hamming codes and BCH codes [8]–[13]. Such modified schemes will be referred to as generalized LDPC (GLDPC) codes in this paper. With more powerful (compared with simple SPC codes) block codes involved, GLDPC codes have many potential advantages, including improved performance in very noisy channels (i.e., low rate applications), fast convergence speed and low error floor.

However, the design of a capacity approaching GLDPC code remains a quite difficult issue. This may be attributed to two factors. First, since more complex codes are used at the check nodes, more complex *a posteriori* probability (APP) decoders are necessary. Thus decoding complexity may become a serious concern. Second, the code optimization techniques based on density evolution [5], [6] or extrinsic mutual information transfer (EXIT) charts [14] for the standard LDPC codes may not be effective for GLDPC codes (as explained later in Section IV). Note that the techniques studied in [4]–[7] are most useful in improving asymptotic (code length $\rightarrow \infty$) performance. On the other hand, combinatorial design approaches [15]–[17] are powerful in improving error floor performance.

The focus of this paper is on asymptotic performance for very low rate applications. We study a family of GLDPC codes that use Hadamard codes at the check nodes. The complexity issue mentioned above is solved by employing the fast APP decoding technique for the Hadamard codes [18]. With regard to the code optimization issue, we analyze the difficulties in directly applying density evolution and EXIT chart based design techniques to GLDPC codes. Such difficulties include mismatching of EXIT functions and the halting of mutual information (or density) evolution caused by degree-1 variables nodes. We propose a solution to both difficulties by introducing the so-called LDPC-Hadamard code structure. We demonstrate the optimization procedure for LDPC-Hadamard codes using the low-cost EXIT chart technique. (We expect that a similar procedure using

Manuscript received September 27, 2005; revised November 18, 2006. This work was supported in part by the U.S. National Science Foundation under Grant CCR-0207550. The material in this paper was presented in part at the 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, September 2005.

G. Yue is with the NEC Laboratories America, Inc., Princeton, NJ 08540 USA.

L. Ping is with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong (e-mail: eeliping@cityu.edu.hk).

X. Wang is with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA (e-mail: wangx@ee.columbia.edu).

Communicated by M. P. C. Fossorier, Associate Editor for Coding Techniques.

Color version of Fig. 4 is available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2006.890694

density evolution can be developed, although we do not elaborate this alternative approach in the present paper.)

We will focus on low-rate applications and demonstrate an LDPC-Hadamard code with large block length that achieves 10^{-5} BER at $E_b/N_0 = -1.44$ dB. This represents an improvement toward the Shannon limit, the ultimate Shannon capacity being $E_b/N_0 = -1.592$ dB for the codes with rates approaching zero [18]–[20], with a remaining gap of only 0.15 dB. (For comparison, the rate-0.019 turbo-Hadamard code reported in [18] achieves a bit error rate (BER) of 10^{-5} at $E_b/N_0 = -1.2$ dB, yielding a gap of about 0.39 dB from the Shannnon limit).

A potential application of the proposed code is in code-division multiple-access (CDMA) systems. In [21], it is shown that the capacity of a random waveform CDMA system can be achieved when very low-rate capacity-achieving codes are used for both error control and bandwidth expansion. Practical transmission and detection methods have been developed in [22] for such a low-rate coded CDMA scheme. The related spectrum efficiency is related to the performance of the low-rate codes involved, and so the codes proposed in this paper can be useful. Another possible application of such low-rate codes is for short-range wireless communications using the ultrawide-band (UWB) technologies [23]. The key for a UWB system is the capability of providing reliable data transmission over a very wide bandwidth with low power consumption to meet regulatory specifications. The coding rate of a UWB system can be low but its power efficiency must be high. The codes discussed in this paper appear promising for such application. A possible third application of the proposed codes is satellite or deep space communications where transmission power is strictly limited but bandwidth usage is more relaxed. Devices that are battery powered may have tight constraints on power consumption and/or size. This may limit the use of complex decoders (and encoders) and/or the maximum allowable iterations. We will show that the encoder complexity of the proposed codes is simple. It is then well suited to the downlink of satellite transmissions.

The remainder of this paper is organized as follows. In Section II, we introduce the LDPC-Hadamard codes and graph representations. Two different Hadamard codes, systematic and nonsystematic codes, are considered for replacing the SPC at check nodes. We present a message-passing decoding algorithm employing low-complexity APP-FHT in Section III. In Section IV, we consider the analysis and optimization of LDPC-Hadamard codes. Numerical results are presented in Section V and the conclusions are summarized in Section VI.

II. Code Structure

A. Hadamard Codes

An $n \times n$ $(n = 2^r)$ Hadamard matrix $\boldsymbol{H}_n$ over $\{+1, -1\}$ can be constructed recursively as

$$\boldsymbol{H}_n = \begin{bmatrix} +\boldsymbol{H}_{\frac{n}{2}} & +\boldsymbol{H}_{\frac{n}{2}} \\ +\boldsymbol{H}_{\frac{n}{2}} & -\boldsymbol{H}_{\frac{n}{2}} \end{bmatrix} \text{ with } \boldsymbol{H}_1 = [+1]. \tag{1}$$

A length-2^r Hadamard codeword set is then formed by the columns of the biorthogonal Hadamard matrix $\pm\boldsymbol{H}_n$, denoted by $\{\pm\boldsymbol{h}^j : j = 0, 1, \ldots, 2^r - 1\}$, in a binary $\{0, 1\}$ form with $+1 \rightarrow 0$ and $-1 \rightarrow 1$. We call r the order of the Hadamard code. Each Hadamard codeword carries $(r + 1)$ bits of information. In systematic encoding of a length-2^r Hadamard code, the information bit positions are indicated by indexes $\{0, 1, 2, 4, \ldots, 2^{r-1}\}$. The other bits are parity bits. Denote $\mathcal{H}$ as a Hadamard encoder. Then, given the input information sequence $b(0), b(1), \ldots, b(r)$, the Hadamard encoding can be represented by

$$\boldsymbol{c} = \mathcal{H}\left(b(0), b(1), \ldots, b(r)\right) \tag{2}$$

where $\boldsymbol{c} = [c(0), c(1), \ldots, c(2^r - 1)]$ is the resulting codeword. Clearly, $\boldsymbol{c}$ is a column of either $+\boldsymbol{H}_n$ or $-\boldsymbol{H}_n$. For *systematic* Hadamard codes, we have $c(0) = b(0)$ and $c(2^{j-1}) = b(j), j = 1, \ldots, r$. Hereafter, for simplicity, we will drop the subscript n in $\boldsymbol{H}_n$.

Based on the formation of the Hadamard matrix in (1), we then obtain the Hadamard codewords with order r recursively. When $r = 2$, it is easy to verify that the Hadamard code is actually a $(4, 3)$ single parity-check (SPC) code with the generator matrix given by

$$\boldsymbol{G}_{H,r=2} = \begin{bmatrix} 1 & & & 1 \\ & 1 & & 1 \\ & & 1 & 1 \end{bmatrix}. \tag{3}$$

Define $\boldsymbol{b}_r = [b(0), b(1), \ldots, b(r)]$. The Hadamard codeword of $r = 2$ is then obtained by

$$\boldsymbol{c}_{r=2} = \boldsymbol{b}_2 \boldsymbol{G}_{H,r=2}. \tag{4}$$

Given the Hadamard codeword $\boldsymbol{c}_r$ obtained from $\boldsymbol{b}_r$, the Hadamard codewords of order $r + 1$ can be obtained by

$$\boldsymbol{c}_{r+1} = [\,\boldsymbol{c}_r, \quad b(r+1) \oplus b(0) \oplus \boldsymbol{c}_r\,] \tag{5}$$

where $\oplus$ denotes the binary addition. Therefore, we can obtain Hadamard codeword recursively by (5) starting from $r = 2$. Similarly, we can form the generator matrix of the Hadamard codes with order $r + 1$, $\boldsymbol{G}_{H,r+1}$, recursively, given by

$$\boldsymbol{G}_{H,r+1} = \begin{bmatrix} \boldsymbol{G}_{H,r} & \boldsymbol{J}_r \oplus \boldsymbol{G}_{H,r} \\ 0\,0\,\ldots\,0, & 1\,1\,\ldots\,1 \end{bmatrix} \tag{6}$$

where $\boldsymbol{J}_r$ has the same size as $\boldsymbol{G}_{H,r}$, given by

$$\boldsymbol{J}_r = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ 0 & 0 & \ldots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix}. \tag{7}$$

Obviously, the complexity of the recursive Hadamard encoder in (5) is less than directly encoding using the generator matrix, i.e., $\boldsymbol{c}_r = \boldsymbol{b}_r \boldsymbol{G}_{H,r}$.

We can also construct a *nonsystematic* Hadamard code as follows. First, let

$$\tilde{b}(j) = b(0) \oplus b(j), \quad j = 1, 2, \ldots, r. \tag{8}$$

Then, we perform the Hadamard encoding for $\{\tilde{b}(j)\}$, i.e.

$$\boldsymbol{c} = \mathcal{H}\left(b(0), \tilde{b}(1), \tilde{b}(2), \ldots, \tilde{b}(r)\right) \tag{9}$$

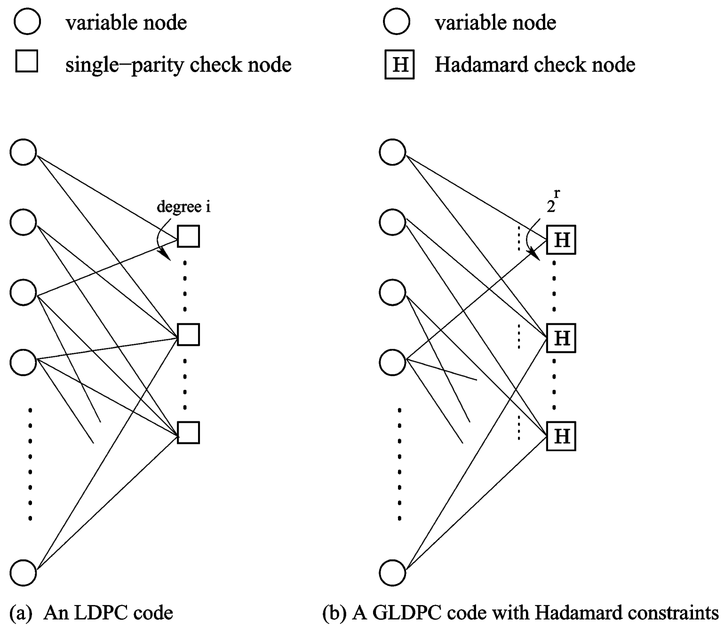


Fig. 1. Bipartite Tanner graph representations of a standard LDPC code and a GLDPC code with Hadamard constraints.

where $c(0) = b(0)$, $c(2^{j-1}) = \tilde{b}(j)$, $j = 1, \ldots, r$. We will discuss the application of nonsystematic Hadamard codes in Sections III and IV.

B. LDPC and GLDPC Codes

We first consider standard LDPC codes. An LDPC code can be represented by a bipartite Tanner graph, illustrated on the left part of Fig. 1. The left nodes are called *variable* nodes and the right nodes are called *check* nodes. A variable node with degree-j is a $(j,1)$ repetition code, while a check node with degree-j is a $(j, j-1)$ SPC code. An LDPC code can then be specified by a graph with the edges connecting the left and right nodes that satisfy both repetition code constraints at the variable nodes and SPC constraints at the check nodes.

Based on the bipartite graph in Fig. 1, a GLDPC code can be defined by replacing the right SPC codes in an LDPC code with any other block codes, for example, Hamming codes or BCH codes [11], [12], [24]. In this paper, we consider GLDPC codes using Hadamard codes at the right check nodes. Such codes have good performance in very noisy channels, as demonstrated later. As illustrated in Fig. 1(b), we now call the right nodes *Hadamard check* nodes. The Hadamard check nodes can either be systematic Hadamard codes or nonsystematic Hadamard codes, which will be discussed further in Section IV.

Note that we may also adopt general block codes for the left nodes (other than the repetition codes, as with a standard LDPC code) in a GLDPC code [25]. However, we will not consider such generalization in this paper. Also note that other equivalent low rate codes such as the balanced incomplete block (BIB) designs [26] can also be used equivalently at check nodes. We expect that the EXIT curve matching problem and its solution is applicable to other block coding constraint at the check nodes. However, due to space limitation, we will focus on Hadamard constraints and will not consider general cases in detail in this paper.

C. Exit Chart Analysis for LDPC Codes

The EXIT chart is a useful tool for analyzing the convergence behavior of iterative processes and code optimization [12], [14], [27]–[29]. We first define $I_Y = I(X;Y)$ as the mutual information between the transmitted bits X and the extrinsic messages Y. Assume X is binary mapped. Given the conditional probability density functions (pdf) $f_Y(y|X=x)$, $x = \pm 1$, we then have [14]

$$I_Y = \frac{1}{2}\sum_{x=\pm 1}\int_{-\infty}^{+\infty} f_Y(y|X=x)\log_2 \times \frac{2f_Y(y|X=x)}{f_Y(y|X=-1)+f_Y(y|X=1)}dy. \quad (10)$$

Assuming Y is a symmetric Gaussian distributed, i.e., $f_Y(y|X=x) = \mathcal{N}(\mu x, 2\mu)$, we then obtain

$$I_Y = 1 - \int_{-\infty}^{\infty} \frac{e^{-(y-\mu)^2/4\mu}}{\sqrt{4\pi\mu}} \log_2(1+e^{-y})dy. \quad (11)$$

We define a function

$$J(m) \triangleq I_Y(\mu = m). \quad (12)$$

The iterative decoding processes can be represented by an EXIT chart with two EXIT curves. One is for the decoding of repetition codes at the variable nodes, the other is for the code constraint at the check node. Define $I_{A,V}$ and $I_{E,V}$ as the input and output extrinsic mutual information of the decoding process for variable nodes, respectively. Similarly define $I_{A,C}$ and $I_{E,C}$ for the single parity check nodes. In [27], it is shown that to approach capacity on erasure channels for an LDPC code, the EXIT curve of the variable nodes must match with the EXIT curve of the check nodes.

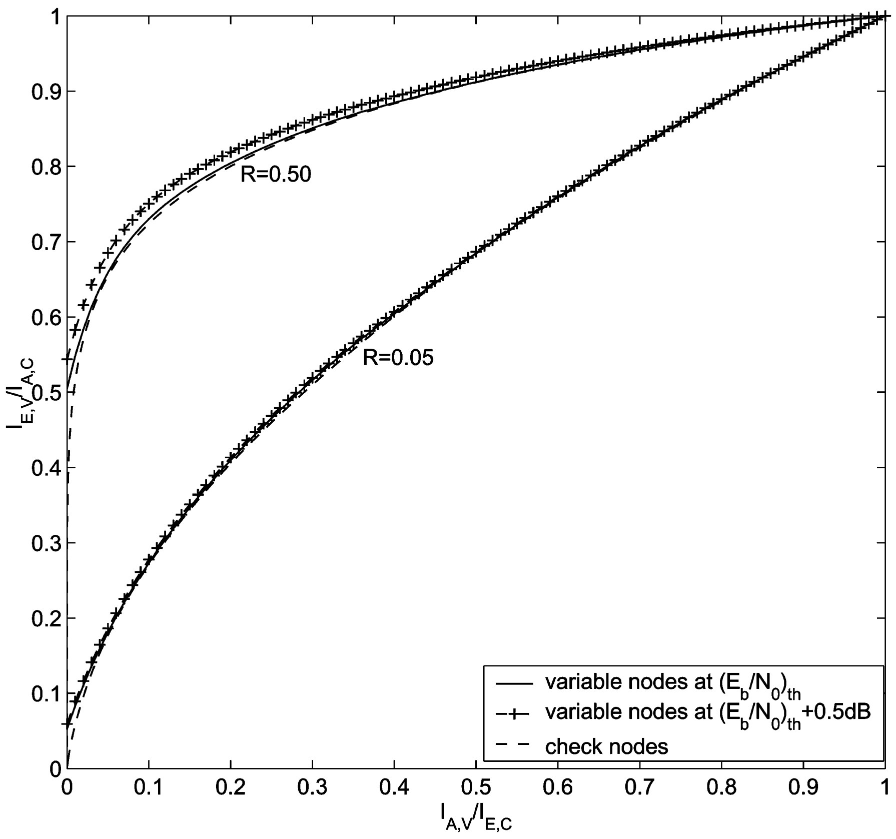


Fig. 2. EXIT curves of variable nodes and check nodes of rate-0.5 and rate-0.05 irregular LDPC codes.

D. The Difficulty in Low-Rate LDPC Code Design

We now, based on EXIT chart analysis, briefly explain the difficulty in obtaining good low-rate LDPC codes. Fig. 2 illustrates EXIT curves of the rate-0.5 and rate-0.05 irregular LDPC codes using SPC codes at the check node. The optimized ensemble profiles are obtained from [30]. The threshold from density evolution, $(E_b/N_0)_{th}$, is 0.25 dB for the rate-0.5 code with maximum left degree being 30. The gap to Shannon capacity is 0.07 dB. For the rate-0.05 irregular code, $(E_b/N_0)_{th} = -1.19$ dB, and the gap to Shannon capacity is 0.25 dB.

The above designs are based on idealized assumptions of infinite code length and unlimited number of iterations. In practice, we have to use finite code length and limited number of iterations and the required E_b/N_0 values will be higher than the thresholds for both codes to work properly. A qualitative discussion on this issue is outlined below.

For each code ensemble, we consider two E_b/N_0 values, i.e., $(E_b/N_0)_{th}$ and $(E_b/N_0)_{th}$+0.5 dB. It is seen from Fig. 2 that for the rate-0.5 irregular LDPC codes, a 0.5 dB increase changes the variable node curve so much that a large iterative decoding tunnel exists between the variable and the check curves, indicating that, due to such SNR increase, a code with a finite length may still be able to work well. However, the 0.5 dB SNR barely changes the variable curve for the rate-0.05 irregular LDPC codes. The decoding tunnel is still very narrow. This implies convergence difficulty for such a low-rate LDPC code even after increasing the SNR value by 0.5 dB from the threshold.

The above observation is based on simulation results. The following discussion may provide more insight into the problem. The EXIT function for the variable nodes of degree d_v is given by

$$I_{E,V}(I_{A,V}, I_{ch}, d_v) = J\left((d_v - 1)J^{-1}(I_{A,V}) + J^{-1}(I_{ch})\right) \quad (13)$$

where I_{ch} is the extrinsic mutual information from the channel. For the AWGN channel, we have $J^{-1}(I_{ch}) = 4R\frac{E_b}{N_0}$. The area under the variable EXIT curve is then given by

$$\begin{aligned}\mathcal{A}\left(\frac{E_b}{N_0}, R, d_v\right) &= \int_0^1 I_{E,V}(I_{A,V}, I_{ch}, d_v)dI_{A,V} \\ &= \int_0^1 J\left((d_v-1)J^{-1}(I_{A,V}) + 4R\frac{E_b}{N_0}\right) \\ &\quad \times dI_{A,V}. \end{aligned} \quad (14)$$

Taking the derivative over E_b/N_0, we obtain

$$\begin{aligned}&\mathcal{A}'\left(\frac{E_b}{N_0}, R, d_v\right) \\ &= 4R\int_0^1 J'\left((d_v-1)J^{-1}(I_{A,V}) + 4R\frac{E_b}{N_0}\right) dI_{A,V}. \end{aligned} \quad (15)$$

Clearly, the area change of EXIT curve of variable nodes is a function of coding rate R. We are interested in capacity ap-

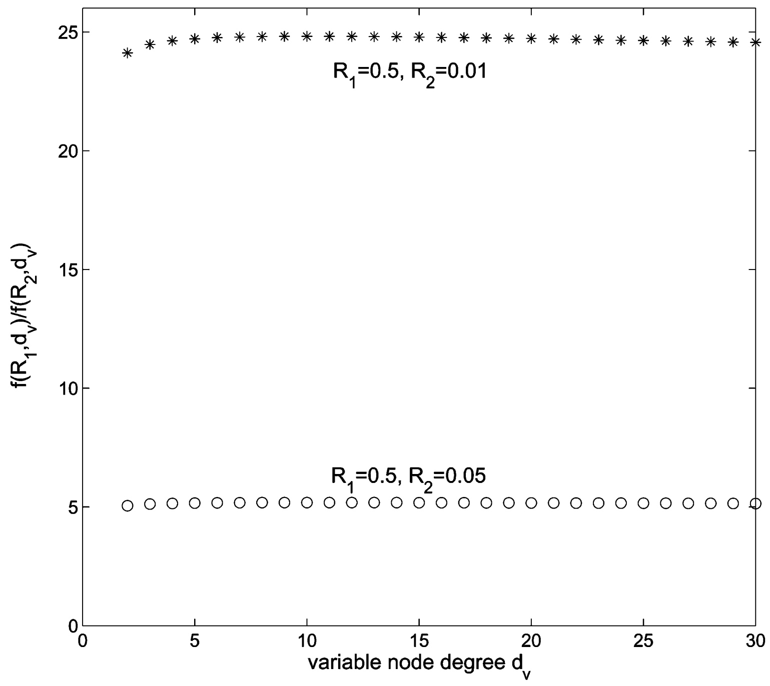


Fig. 3. Ratio of the area change under the variable EXIT curve for two different rates.

proaching codes and so we will examine (15) at $\frac{E_b}{N_0} = \frac{E_b}{N_0}\Big|_{cap}$ where $\frac{E_b}{N_0}\Big|_{cap}(R)$ is capacity limit for code rate R. Define

$$f(R, d_v) \triangleq \mathcal{A}'\left(\frac{E_b}{N_0}\Big|_{cap}(R), R, d_v\right). \tag{16}$$

We then compare the area change for two different rates using $f(R_1, d_v)/f(R_2, d_v)$, as shown in Fig. 3. We find that the area change rate over E_b/N_0 for $R = 0.5$ is around 5 times of that for $R = 0.05$, and 24 times of that for $R = 0.01$. The results are quite uniform for various d_v. This explains why we see little change on the variable EXIT curve with a 0.5 dB increase from the threshold value for a low rate code. It is worth noting that, since the capacity limit for low rate codes is smaller than that for medium rate, same value increase in decibels results smaller change on the absolute value of E_b/N_0 at lower rate.

The above discussion indicates that lower rate LDPC codes require larger SNR increase from the decoding threshold to obtain similar conditions regarding decoding tunnels (or to obtain the same area of the decoding tunnel) than their higher rate counterparts, as indicated by the simulation results in Fig. 2.

III. The Decoder Structure

A. Generalized Message-Passing Algorithm

Similarly to the LDPC decoding algorithm, we can decode a GLDPC code iteratively by applying generalized message-passing based on the Tanner graph representation. The extrinsic messages are iteratively exchanged between the left nodes and the right nodes as the input and output of the decoders. Consider a GLDPC code in the presence of AWGN channel distortion and define the log-likelihood ratios (LLRs) of the bits as the *messages*. Denote $u_{c\to v,j}$ as the extrinsic message output from a check node passed along the jth edge to a variable node and $v_{v\to c,j}$ as the extrinsic message output from a variable node passed along the jth edge to a check node. Denote $\lambda_{v,n}$ as the message of the nth variable node from the AWGN channel. Define $\mathcal{V}_n$ as the set of the edges that are connected to the nth variable node and $\mathcal{U}_m$ as the set of the edges that are connected to the mth check node. Then based on the message-passing algorithm [7], the extrinsic messages updated from the ℓth stage to the $(\ell+1)$-th stage can be obtained by

$$\begin{aligned} v^{\ell}_{v\to c,k} &= \mathcal{C}^{-1}_{v,n}\left(\left\{u^{\ell}_{c\to v,j} | j \in \mathcal{V}_n\right\}, \lambda_{v,n}\right) - u^{\ell}_{c\to v,k} \\ k &\in \mathcal{V}_n, \quad n = 1, \ldots, N \\ u^{\ell+1}_{c\to v,k} &= \mathcal{C}^{-1}_{c,m}\left(\left\{v^{\ell}_{v\to c,j} | j \in \mathcal{U}_m\right\}\right) - v^{\ell}_{v\to c,k} \\ k &\in \mathcal{U}_m, \quad m = 1, \ldots, M \end{aligned} \tag{17}$$

with

$$u^{0}_{c\to v,j} = 0, \quad \forall\, j \tag{18}$$

where $\mathcal{C}^{-1}_{v,n}$ denotes the APP decoding process of the nth variable node and $\mathcal{C}^{-1}_{c,m}$ denotes the APP decoding process of the mth check node. For the LDPC-Hadamard codes, the iterative decoding process is then given by

$$v^{\ell}_{v\to c,k} = \sum_{j\in\mathcal{V}_n, j\neq k} u^{\ell}_{c\to v,j} + \lambda_{v,n}, \quad k \in \mathcal{V}_n$$
$$n = 1, \ldots, N \tag{19}$$

$$\begin{aligned} u^{\ell+1}_{c\to v,k} &= \mathcal{H}^{-1}_{m,k}\left(v^{\ell}_{v\to c,j^{(1)}}, \ldots, v^{\ell}_{v\to c,j^{(i)}}, \ldots, v^{\ell}_{v\to c,j^{(d_m)}}\right) \\ &\quad - v^{\ell}_{v\to c,k} \\ j^{(i)} &\in \mathcal{U}_m, \quad k \in \mathcal{U}_m, \; m = 1, \ldots, M \end{aligned} \tag{20}$$

where $\mathcal{H}^{-1}$ denotes the APP Hadamard decoding process with the inputs being the extrinsic LLRs of all the bits of the

Hadamard code and the outputs being the APP LLRs; $\mathcal{H}_{m,k}^{-1}$ denotes the output LLR along the kth edge from the decoding process of the mth Hadamard check node; and d_m is number of edges connected to the mth Hadamard check nodes with order r_m, i.e., $d_m = 2^{r_m} = |\mathcal{U}_m|$.

B. Fast APP Decoding of Hadamard Codes

As mentioned above, during each decoding iteration APP decoding is performed at the check nodes and variable nodes. Therefore, fast APP decoding algorithms for general code constraints at both nodes are essential to reduce the GLDPC decoding complexity. Fast APP algorithms are available for SPC, Reed-Muller, and Hadamard codes [18], [31]. It is also possible to devise fast APP decoding for Hamming codes based on minimum trellis structures [32]. We briefly outline the APP decoding of Hadamard codes and its fast algorithm.

Denote $\boldsymbol{x} = [x(0), x(1), \ldots, x(2^r - 1)]^T$ as the received sequence of Hadamard code bits $\boldsymbol{c}$ corrupted by AWGN with variance σ^2. Given the *a priori* LLRs of the Hadamard coded bits, $\boldsymbol{L}_{apr} = [L_{apr}(0), L_{apr}(1), \ldots, L_{apr}(2^r - 1)]^T$, the LLRs from the APP decoding over $\{+1, -1\}$ are obtained by [18]

$$L_{app}(i) = \log \frac{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j) = \pm 1\}} \gamma\left(\pm \boldsymbol{h}^j\right)}{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j) = \mp 1\}} \gamma\left(\pm \boldsymbol{h}^j\right)} \tag{21}$$

where

$$\gamma\left(\pm \boldsymbol{h}^j\right) \triangleq \exp\left(\frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \frac{2\boldsymbol{x}}{\sigma^2} + \boldsymbol{L}_{\text{apr}}\right\rangle\right) \tag{22}$$

and $\langle \cdot, \cdot \rangle$ denotes the inner product. The extrinsic LLR is then given by

$$L_{\text{ext}}(i) = L_{\text{app}}(i) - L_{\text{apr}}(i). \tag{23}$$

Some details are given in Appendix A.

Direct calculation of the LLR in (21) has a high computational complexity, i.e., $\mathcal{O}(2^{2r})$. The fast Hadamard transform (FHT) [18] can be employed to reduce the complexity of the inner product in (21) based on the butterfly graph of the Hadamard matrix. We first segment $\boldsymbol{x}$ into two subvectors, i.e., $\boldsymbol{x} = \left[\boldsymbol{x}_1^T, \boldsymbol{x}_2^T\right]^T$. To compute $y(j) = \langle \boldsymbol{h}^j, \boldsymbol{x} \rangle$ for $j = 0, 1, \ldots, 2^r - 1$, we have

$$\boldsymbol{y} = \boldsymbol{H}_n \boldsymbol{x} = \begin{bmatrix} +\boldsymbol{I}_{\frac{n}{2}} & +\boldsymbol{I}_{\frac{n}{2}} \\ +\boldsymbol{I}_{\frac{n}{2}} & -\boldsymbol{I}_{\frac{n}{2}} \end{bmatrix} \begin{bmatrix} \boldsymbol{H}_{\frac{n}{2}} \boldsymbol{x}_1 \\ \boldsymbol{H}_{\frac{n}{2}} \boldsymbol{x}_2 \end{bmatrix} \tag{24}$$

where $\boldsymbol{y} = [y(0), \ldots, y(2^r - 1)]^T$, $\boldsymbol{I}_{\frac{n}{2}}$ is an identity matrix. It is seen that we can recursively factorize $\boldsymbol{H}_{\frac{n}{2}} \boldsymbol{x}_1$ and $\boldsymbol{H}_{\frac{n}{2}} \boldsymbol{x}_2$ in the same way and stop at $\boldsymbol{H}_1 x(j)$, $j = 0, 1, \ldots, 2^r - 1$. It can be seen that the process of computing $\boldsymbol{y}$ is similar to the fast Fourier transform (FFT) computation.

The summations in the numerator and denominator in (21) for computing $L_{app}(i), i = 0, 1, \ldots, 2^r - 1$, can be efficiently computed by the APP-FHT algorithm [18]. With the APP-FHT algorithm, the complexity for calculating (21) is reduced to $\mathcal{O}(r2^r)$.

C. App Decoding of Nonsystematic Hadamard Codes

With the APP-FHT algorithm, we can easily obtain the APP LLRs for each code bit in a systematic Hadamard codeword. For nonsystematic Hadamard codes, directly applying the APP-FHT algorithm will only yield the APP LLR for $\tilde{b}_j$. In order to get the APP decoding of information bits b_j using the APP-FHT algorithm, we can employ the following data preprocessing on the input LLRs before decoding.

Let us compare the nonsystematic Hadamard codewords with the corresponding systematic Hadamard codewords of the same order r. When $b_0 = 0$, the codeword set of the nonsystematic Hadamard codes is exactly the same as that of the systematic codes. Those codewords are formed from the positive Hadamard matrix $\boldsymbol{H}$, i.e., $+\boldsymbol{h}^j$, $j = 0, \ldots, 2^r - 1$. When $b_0 = 1$, the nonsystematic codeword with the input $b_1 b_2 \ldots b_r$ is equal to the systematic codeword $\overline{b}_1 \overline{b}_2 \ldots \overline{b}_r$, i.e., the binary complementary sequence of input $b_1 b_2 \ldots b_r$. Therefore, the APP decoding of nonsystematic Hadamard code can be obtained by (21) with $\gamma(\pm \boldsymbol{h}^j)$ replaced by pre-processed $\gamma'(\pm \boldsymbol{h}^j)$ following the rules below:

- $\gamma'(+\boldsymbol{h}^j) = \gamma(+\boldsymbol{h}^j)$, $j = 0, \ldots, 2^r - 1$.
- $\gamma'(-\boldsymbol{h}^j) = \gamma(-\boldsymbol{h}^k)$, where j is the binary complementary of k.

It should be noted that the above preprocessing is necessary only for the information bits, $b_1, b_2, \ldots, b_r$, i.e., $L_{\text{app}}(2^j)$, $j = 0, \ldots, r - 1$, since only these bits have undergone special treatments in the encoder, (see (9)). We can obtain other $L_{\text{app}}(i)$ by directly applying APP-FHT decoding algorithm in the decoder of the nonsystematic Hadamard codes without preprocessing.

IV. GLDPC Code Design and Analysis

The density evolution method [6] has been successfully applied to the analysis and design for standard LDPC codes. A closely related method is the EXIT chart technique [29], which is a low complexity one dimensional approach to characterize the iterative detection process. In this section, we discuss two difficulties in applying the EXIT chart technique to GLDPC code design, i.e., the problems caused by mismatching of the EXIT functions and the existence of degree-1 nodes. We then proceed to devise solutions to these two problems.

A. The EXIT Function Mismatching Problem

For LDPC codes, the code ensemble $\left\{\lambda_j^L, \rho_j^L\right\}$ can be optimized to have asymptotic performance approaching the channel capacity [6], [33]. However, it is still an open question as to whether GLDPC codes can be optimized to approach capacity with the check nodes formed by block codes other than SPC codes. The answer seems negative if the standard LDPC structure is directly adopted. We will explain the reason below and explore potential solutions.

In this paper, we will assume that repetition codes are always used at the variable nodes. As demonstrated in [6] for LDPC codes, the SPC codes can be used at the check nodes in order to approach capacity. Next, we investigate the performance of GLDPC codes with Hadamard codes at the check nodes using EXIT chart analysis.

For the two EXIT curves, the curve for the variable nodes is the same as that discussed in Section II-D. The other one for the check nodes can be generated based on simulation involving Hadamard codes. Define $I_{A,H}$ and $I_{E,H}$ for the Hadamard check nodes. The EXIT functions for the variable nodes of degree-j and Hadamard check nodes of order r are denoted by $\mathcal{G}_{V,j}\left(I_{A,V}, j, \frac{E_s}{N_0}\right)$ and $\mathcal{G}_{H,r}(I_{A,H}, r)$, respectively. The values of $\mathcal{G}_{V,j}\left(I_{A,V}, j, \frac{E_s}{N_0}\right)$ and $\mathcal{G}_{H,r}(I_{A,H}, r)$ can be computed from the output extrinsic conditional pdfs by (10). For binary codes, the EXIT curve of the extrinsic messages from the codes consisting of different component codes, e.g., the SPC codes with deferent degrees, is an average of the EXIT curves of its component codes [27], [34]. We specify the LDPC-Hadamard code ensemble by profiles of $\{\lambda_j\}_{j=1}^{d_{v_{\max}}}$ and $\{\rho_r\}_{r=1}^{d_{r_{\max}}}$, where r denotes the order of the Hadamard check nodes. The average output extrinsic mutual information is then given by

$$I_{E,V} \triangleq \mathcal{G}_V\left(I_{A,V}, \frac{E_s}{N_0}\right) = \sum_j \lambda_j \mathcal{G}_{V,j}\left(I_{A,V}, j, \frac{E_s}{N_0}\right) \quad (25)$$

$$I_{E,H} \triangleq \mathcal{G}_H(I_{A,H}) = \sum_r \rho_r \mathcal{G}_{H,r}(I_{A,H}, r). \quad (26)$$

The conditional pdf of the input LLRs from AWGN channels at the variable nodes is symmetric Gaussian with mean $m_{\mathrm{ch}} = \frac{4E_s}{N_0}$. Since the variable nodes are repetition codes, from (19), we then have [29]

$$\mathcal{G}_{V,j}\left(I_{A,V}, j, \frac{E_s}{N_0}\right) = J\left((j-1)J^{-1}(I_{A,V}) + m_{\mathrm{ch}}\right). \quad (27)$$

The EXIT functions of SPC check nodes can be approximated as [29]

$$\begin{aligned} I_{E,C} \triangleq \mathcal{G}_C(I_{A,C}) &= \sum_j \rho_j \mathcal{G}_{C,j}(I_{A,C}, j) \\ &\approx \sum_j \rho_j \left(1 - J\left((j-1)J^{-1}(1 - I_{A,C})\right)\right) \end{aligned} \quad (28)$$

where $I_{A,C}$ and $I_{E,C}$ are the input and output extrinsic mutual information of SPC decoding.

Fig. 4 shows the EXIT curves for GLDPC codes in AWGN channels. The upper plot illustrates the EXIT curves for variable nodes. The lower one illustrates the EXIT curves for SPC and Hadamard check nodes. The variable transfer curves are obtained from (27) with $E_s/N_0 = -11.28$ dB, the capacity limit for a rate-0.1 channel code. The EXIT curves for SPC are obtained from (28). The Hadamard check node curves are obtained from the APP Hadamard decoding in (21) and (23) using the Monte Carlo method. The input of the EXIT curves for the variable nodes is along the x-axis and the output is along the y-axis, and vice versa for the check node curves.

In [27], it is shown that to approach capacity on erasure channels for an LDPC code, the EXIT curve of the variable nodes must match with the EXIT curve of the check nodes. We conjecture that this is also true for GLDPC codes on the AWGN channel. Based on this conjecture, we now examine the EXIT curves in Fig. 4. The set of curves for repetition codes are concave and do not have saddle points. The EXIT curves for SPC codes are also concave and so they may be optimized for matching. (Note: These curves are drawn with the abscissa as output. Therefore, the curves are concave but the input-output functions are actually convex.) However, the EXIT curves for the Hadamard codes are neither concave or convex. They have saddle points (except for $r = 2$, which is equivalent to a length-4 SPC code). Also, it is easy to verify that any linear combinations of the curves for Hadamard codes of $r \geq 3$ have saddle points. Thus we cannot use these codes for matching (except using only $r = 2$ codes which results in standard LDPC codes). Therefore, the EXIT curves of the variable and Hadamard check nodes cannot be optimized to match with each other. This constitutes the first problem in designing optimized GLDPC codes.

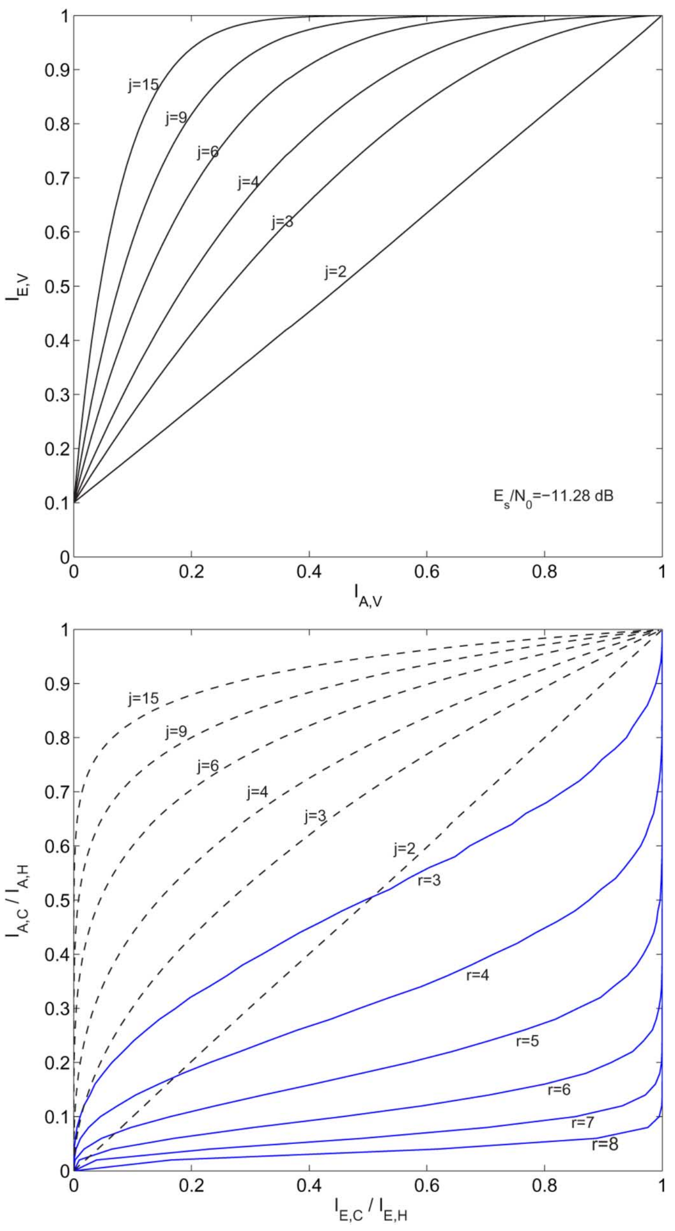


Fig. 4. EXIT curves for GLDPC codes. Upper plot: EXIT curves for variable nodes of degree-j. $E_s/N_0 = -11.28$ dB. Bottom plot: EXIT curves for check nodes. Dashed lines: SPC nodes of degree-j; solid lines: Hadamard check nodes of order-r.

*B. The Degree-*1 *Nodes Problem*

We now explain another problem for GLDPC design which is caused by degree-1 nodes.

1) Code Rate of GLDPC Codes: Consider a GLDPC code C with N variable nodes formed by N repetition codes, and M check nodes formed by M block codes. Define, respectively, $n_{c,i}$, $k_{c,i}$, and $r_{c,i}$, $i = 1, \ldots, M$, for the block codes at the check nodes. Denote $n_{v,i}$, $i = 1, \ldots, N$, as the repetition times (degrees) at the variable nodes. The total number of edges is then

$$n = \sum_{i=1}^{N} n_{v,i} = \sum_{i=1}^{M} n_{c,i}. \tag{29}$$

Each check code contributes $n_{c,i} - k_{c,i}$ constraints. Assuming that these constraints are all independent, the total number of constraints is

$$E = \sum_{i=1}^{M} (n_{c,i} - k_{c,i}). \tag{30}$$

The code rate of GLDPC is then given by

$$R = 1 - \frac{E}{N} = 1 - \frac{\sum_{i=1}^{M} (n_{c,i} - k_{c,i})}{N}. \tag{31}$$

Define the average rates of variable nodes and check nodes as

$$R_v = \frac{N}{n} \tag{32}$$

$$R_c = \frac{1}{n} \sum_{i=1}^{M} k_{c,i}. \tag{33}$$

The code rate of GLDPC in (31) can also be written as

$$R = 1 - \frac{1 - \frac{\sum_{i=1}^{M} k_{c,i}}{\sum_{i=1}^{M} n_{c,i}}}{\frac{N}{\sum_{i=1}^{N} n_{v,i}}} = 1 - \frac{1 - R_c}{R_v}. \tag{34}$$

We assume that in a GLDPC code C, the repetition codes at the left nodes and the block codes at the right nodes are formed from the codes of rate $1/j$ with fraction λ_j and the codes of rate $R_{c,j}$ with fraction ρ_j, respectively. The code rate of GLDPC in (31) can then be written as

$$R = 1 - \frac{\sum_{j=1}^{d_{c\max}} \rho_j (1 - R_{c,j})}{\sum_{j=1}^{d_{v\max}} \lambda_j / j}$$

with

$$\sum_{j=1}^{d_{v\max}} \lambda_j = 1 \qquad \sum_{j=1}^{d_{c\max}} \rho_j = 1. \tag{35}$$

When $R_{c,j} = \frac{j-1}{j}$, we obtain the expression of the LDPC code rate

$$R^L = 1 - \frac{\sum_{j=1}^{d_{c\max}} \rho_j / j}{\sum_{j=1}^{d_{v\max}} \lambda_j / j}. \tag{36}$$

Now we consider a GLDPC code with Hadamard constraints, shown in Fig. 1(b). From (35), the code rate of such a code is given by

$$R = 1 - \frac{\sum_r \rho_r (1 - (r+1)/2^r)}{\sum_j \lambda_j / j}. \tag{37}$$

A Hadamard code with $r = 2$ is equivalent to a single parity-check code of degree-4. If $\rho_2 = 1$, the LDPC-Hadamard code becomes a standard LDPC code with right degree $\rho_4^L = 1$.

*2) The Degree-*1 *Nodes Problem for GLDPC Codes:* Let λ_1 be the fraction of edges connected to the degree-1 nodes in a GLDPC code. Degree-1 nodes represent the bits that are directly connected to check nodes with no repetition. From the GLDPC code rate given in (35), we have

$$\sum_{j=1}^{d_{v\max}} \lambda_j / j = \frac{\sum_{j=1}^{d_{c\max}} \rho_j (1 - R_{c,j})}{1 - R}. \tag{38}$$

Since $\sum_{j=1}^{d_{v\max}} \lambda_j = 1$, we have

$$\sum_{j=1}^{d_{v\max}} \lambda_j / j \leq \lambda_1 + \frac{1 - \lambda_1}{2} = \frac{1 + \lambda_1}{2}. \tag{39}$$

Then

$$\begin{aligned} \lambda_1 &\geq \frac{2 \sum_{j=1}^{d_{c\max}} \rho_j (1 - R_{c,j})}{1 - R} - 1 = \frac{2(1 - R_c)}{1 - R} - 1 \\ &= \frac{1 - 2R_c + R}{1 - R}. \end{aligned} \tag{40}$$

Assuming $0 < R < 1$, we then have the following theorem from (40).

Theorem 4.1: For any GLDPC codes with the left nodes being the repetition codes, if the average code rate at check nodes, R_c, is smaller than $\frac{1}{2}$, then the code contains a nonnegligible portion of degree-1 nodes, i.e., λ_1 is bounded away from 0.

Since $R_c = \sum_j \rho_j R_{c,j} \leq \max_j R_{c,j}$ and $R_c \leq \max_i r_{c,i}$, we have the following corollary.

Corollary 4.1: For any GLDPC codes with the left nodes being the repetition codes, if $\max_j R_{c,j} < \frac{1}{2}$ or $\max_i r_{c,i} < \frac{1}{2}$, λ_1 is bounded away from 0.

For a GLDPC code with Hadamard constraints, when $r \geq 3$, from the code rate given in (37), we have

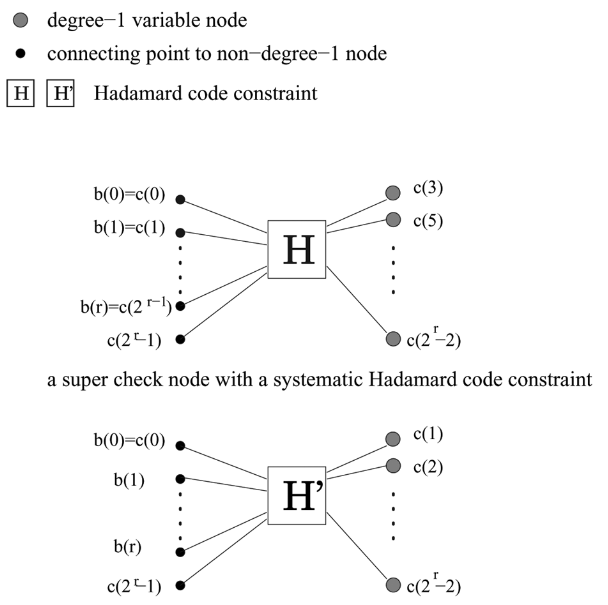
$$R_c = \sum_r \rho_r \frac{r+1}{2^r} \leq \frac{1}{2}. \tag{41}$$

Therefore, based on Theorem 4.1, there exist a nonnegligible portion of degree-1 nodes in such a code. In particular, when the Hadamard check nodes are regular, i.e., all of the Hadamard codes at the check nodes have the same order r, we have λ_1 bounded by

$$\lambda_1 > 1 - 2R_c = 1 - \frac{r+1}{2^{r-1}}. \tag{42}$$

Therefore, for $r = 3, 4, 5, 6, 7, 8, \ldots$, we have $\lambda_1 > 0, \frac{3}{8}, \frac{5}{8}, \frac{25}{32}, \frac{7}{8}, \frac{119}{128}, \ldots$. As r increases, the proportion of degree-1 nodes increases.

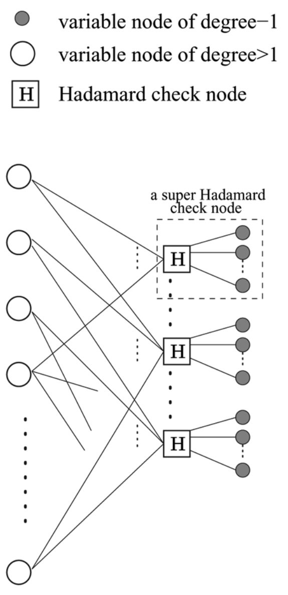
From the standard EXIT chart analysis of LDPC codes, we know that the outbound extrinsic messages of degree-1 nodes will not be updated during the iterative decoding process. This indicates that when λ_1 is bounded away from zero, we cannot use the EXIT chart technique to obtain a threshold where the EXIT information converges to 1 since the outbound extrinsic messages from degree-1 nodes remain unchanged. It should be noted that the discussion above does not mean that a threshold does not exist by iterative decoding. It only states that we cannot



degree−1 variable node
connecting point to non−degree−1 node
H H' Hadamard code constraint
b(0)=c(0)
b(1)=c(1)
b(r)=c(2^{r-1})
c(2^r−1)
c(3)
c(5)
c(2^r−2)
a super check node with a systematic Hadamard code constraint
b(0)=c(0)
b(1)
b(r)
c(2^r−1)
c(1)
c(2)
c(2^r−2)

a super check node with a non−systematic Hadamard code constraint

Fig. 5. Graph representations of super check nodes with systematic and nonsystematic Hamadard code constraints. For an augmented nonsystematic Hadamard code, the degree-1 node should include all coded bits except $c(0)$ and $c(2^r - 1)$.



variable node of degree−1
variable node of degree>1
H Hadamard check node
a super Hadamard check node

Fig. 6. Modified bipartite Tanner graph representation of an LDPC-Hadamard code.

get the threshold, if it exists, by the EXIT chart technique. The reason is that the mutual information used in the EXIT chart method is averaged over all edges. Global convergence in actual decoding can be achieved even if only some of the messages converge.

In summary, the degree-1 problem mentioned above is a problem caused by the combination of the EXIT chart analysis method and the code structure. Some structural changes are necessary before we can successfully apply the EXIT chart method to GLDPC code design.

C. LDPC-Hadamard Code

1) Treatment of Degree-1 Nodes: We now propose a new code structure for LDPC-Hadamard codes to solve the two problems mentioned above. The code graph of the new LDPC-Hadamard code is illustrated in Fig. 6. An LDPC-Hadamard code C is generated using a two-step approach.

- Step 1) Construct a LDPC code C^L without any degree-1 variable nodes.
- Step 2) Use the variable bits connected to each SPC check node in C^L to encode a Hadamard code.

The LDPC-Hadamard code C is then formed by the union of C^L and the Hadamard codes generated in Step 2).

There are two approaches to using the m variable bits connected to a check node in generating the Hadamard codes in Step 2). In a straightforward way, the m variable bits connected to a common check node can be used as the information bits to encode an order-r Hadamard code H with $r = m-1$, following the definition in Section II-A. However, this is not an efficient method, since there is an SPC constraint imposed on these m variable bits, and they are not independent. We can improve efficiency by using the m variable bits to encode an order-r Hadamard code with $r = m - 2$ based on the next theorem.

Theorem 4.2: In a systematic Hadamard code with even order r, and a nonsystematic Hadamard code with any order $r > 1$, we have the following SPC constraint

$$c(2^r - 1) \oplus \sum_{i=0,\dots,r}^{\oplus} b(i) = 0 \tag{43}$$

where $\oplus$ denotes binary addition and $\sum^{\oplus}$ denotes binary summation. Moreover, the SPC constraint in (43) is not satisfied by systematic Hadamard codes with odd r.

The proof is given in Appendix B.

With Theorem 4.2, we can build the check nodes for an LDPC-Hadamard code using a super check node as shown in Fig. 5. We first construct a standard LDPC code C^L. We then replace each check node by a Hadamard check node consisting of a Hadamard check node together with the attached degree-1 nodes. The edges connected to a super check node are the same as those connected to the corresponding check node in C^L, and thus the variable nodes on the left side remain unchanged. The degree-1 nodes inside a super check node represent the extra bits generated during the Hadamard encoding. These degree-1 nodes include all the parity bits of H except $c(2^r - 1)$. (See (43).)

Note: According to the encoding process outlined in Section II-A, the parity positions for a systematic Hadamard code are given by $\{c(3), c(5), c(6), c(7), \ldots\}$, i.e., the integer set obtained by excluding $\{0, 1, 2, 4, \ldots, 2^{r-1}\}$ from $\{0, 1, 2, 3, \ldots, 2^r - 1\}$. For a nonsystematic Hadamard code H, we define an augmented code H' formed by the union of H and $\boldsymbol{b}$, where $\boldsymbol{b}$ is the set of information bits for H. In this way, the parity bits of H' are all the coded bits in H except $b(0)$, which is the only common bit in both H and $\boldsymbol{b}$.

As a brief summary, an LDPC-Hadamard code described above can be viewed as a standard LDPC code cascaded with a special form of Hadamard coding. Graphically, the extra parity bits generated by the Hadamard encoding form the degree-1 variable nodes as each of them is connected only to one super check node. Thus the factor graph for C is obtained simply replacing the check nodes for C^L by super check nodes, as shown in Fig. 6. With the structure shown, the degree-1 nodes carry the information received from the communication channel. However, unlike with standard LDPC codes, we do not include the messages from the degree-1 nodes in calculating the mutual information of the variable nodes during the iterative process since iterative decoding with a degree-1 node results in that node transmitting the same message on every iteration. Note that we DO use the information carried by the degree-1 nodes (obtained from the observations of these bits) in the APP decoding at the super check nodes, thus such APP decoding not only involves the information from the extrinsic channel, but also the information directly from the communication channel. However, the average mutual information for the EXIT curves is only based on the messages from the nondegree-1 nodes. If a code is designed using the EXIT function based optimization as described in Section IV-C2, the EXIT information so generated can converge to 1, as also demonstrated in Section IV-C2.

Note that the common early stopping technique is still applicable here. Recall that an LDPC-Hadamard code is obtained by appending degree-1 Hadamard parity bits to an underlying LDPC core C^L, as shown in Fig. 6. After the APP decoding of the Hadamard constraints at the super check nodes in an iteration, we obtain the messages from the super nodes (on the edges at the left hand side in Fig. 6). We can then apply parity check to the super nodes (ignoring their the appended degree-1 nodes). The iterative decoding stops if the parity check at every super node is satisfied.

The other issue is how the degree-1 nodes affect convergence rate. For generalized LDPC codes, we can replace the SPC constrains at check nodes by different block coding constraints. This may result in different options with more, less or no degree-1 node. (The last option can simply be a standard LDPC code.) The convergence comparison of these different options is a very interesting issue for future research and can be useful in exploring new code structures. However, we will not discuss this issue in detail in this paper, since for the very low rate LDPC-Hadamard codes under consideration, the portion of degree-1 nodes are determined by the rate of the overall code and the order of Hadamard codes involved.

Encoding complexity: In a straightforward way to construct a systematic LDPC code of $N = K + J$, where K and J are the information and parity lengths respectively, we need to convert a parity matrix $\boldsymbol{H}$ into a generator matrix $\boldsymbol{G}$ and store the parity part of $\boldsymbol{G}$ (size $J \times K$). When rate is very low, J is relatively large. Then both computational and storage costs related in $\boldsymbol{G}$ is high. For an LDPC-Hadamard encoder C, the complexity involved in Step 2) described above is very low. (See Section II-A.) For Step 1), we use a generator matrix $\boldsymbol{G}^L$ for the LDPC core C^L. The size of its parity part is $J^L \times K$, where J^L is the number of nondegree-1 parity nodes. Usually $J^L \ll J$ for a low rate code, and so the complexity for encoding C^L is much lower than for directly encoding a standard LDPC code of the same rate and length as C. Efficient LDPC encoding techniques have been extensively studied recently (although whether or not these new techniques can be applied to the low-rate codes still needs careful study). Any fast LDPC code structure that admits low encoder complexity can potentially be employed to construct C^L. Thus we can see the advantage of low-encoding cost for the proposed scheme, which is particularly useful for satellite or deep space communications where devices are battery powered and may have tight constraints on power consumption and size.

2) EXIT Analysis: With the above LDPC-Hadamard structure and the APP decoding procedure described in Section III, the decoding at a super check node C is now based on two pieces of information: the channel outputs as the extrinsic inputs from degree-1 nodes, $\tilde{\boldsymbol{L}}^2_{apr}(i)$, and the extrinsic information from nondegree-1 variable nodes on the left hand side in Fig. 6 connected to C, $\tilde{\boldsymbol{L}}^1_{apr}(i)$. In the EXIT chart analysis presented below, the average mutual information is computed only from the second part, i.e., the extrinsic information from nondegree-1 variable nodes. Empirically, we observed that the EXIT curves so obtained are almost concave, making curve matching possible. Therefore, the modified graph simultaneously solves both the difficulties associated with degree-1 nodes and those associated with curve matching. A rigorous proof of the concavity property of the EXIT curves appears to be a difficult issue, and we can only provide experimental results at this stage.

Since the check nodes in LDPC-Hadamard codes receive information from the communication channel, the EXIT function in (26) now becomes

$$I_{E,H} = \mathcal{G}_H\left(I_{A,H}, \frac{E_s}{N_0}\right) = \sum_r \rho_r \mathcal{G}_{H,r}(I_{A,H}, r, m_{\mathrm{ch}}). \quad (44)$$

The EXIT function for the variable nodes in (25) remains unchanged. The function $J(\cdot)$ in (27) can be approximated by a closed form expression with an exponential function [29]. With this approximation, we can obtain the EXIT curves for the variable nodes directly from calculating (27) and the average output from (25). For EXIT curves of super check nodes, we have to rely on Monte Carlo simulations to obtain the output extrinsic pdf first, and then compute the output extrinsic mutual information.

We first evaluate the component EXIT curves, $\mathcal{G}_{V,j}(I_{A,V}, j, m_{\mathrm{ch}})$ and $\mathcal{G}_{H,r}(I_{A,H}, r, m_{\mathrm{ch}})$, of LDPC-Hadamard codes through an AWGN channel with $E_s/N_0 = -18.51$ dB, i.e., the noise variance $\sigma_n^2 = 35.48$ for unit signal power. This is the Shannon capacity limit for a rate-0.02 code through an AWGN channel with binary inputs. The component EXIT curves for the variable nodes and super check nodes with various degrees or orders are illustrated in the upper part of Fig. 7. The left plot shows the EXIT curves for the decoding at the variable nodes. The right plot show the EXIT curves for the Hadamard decoding at super check nodes for systematic Hadamard codes. It can be seen that for the systematic Hadamard code with even r, the extrinsic mutual

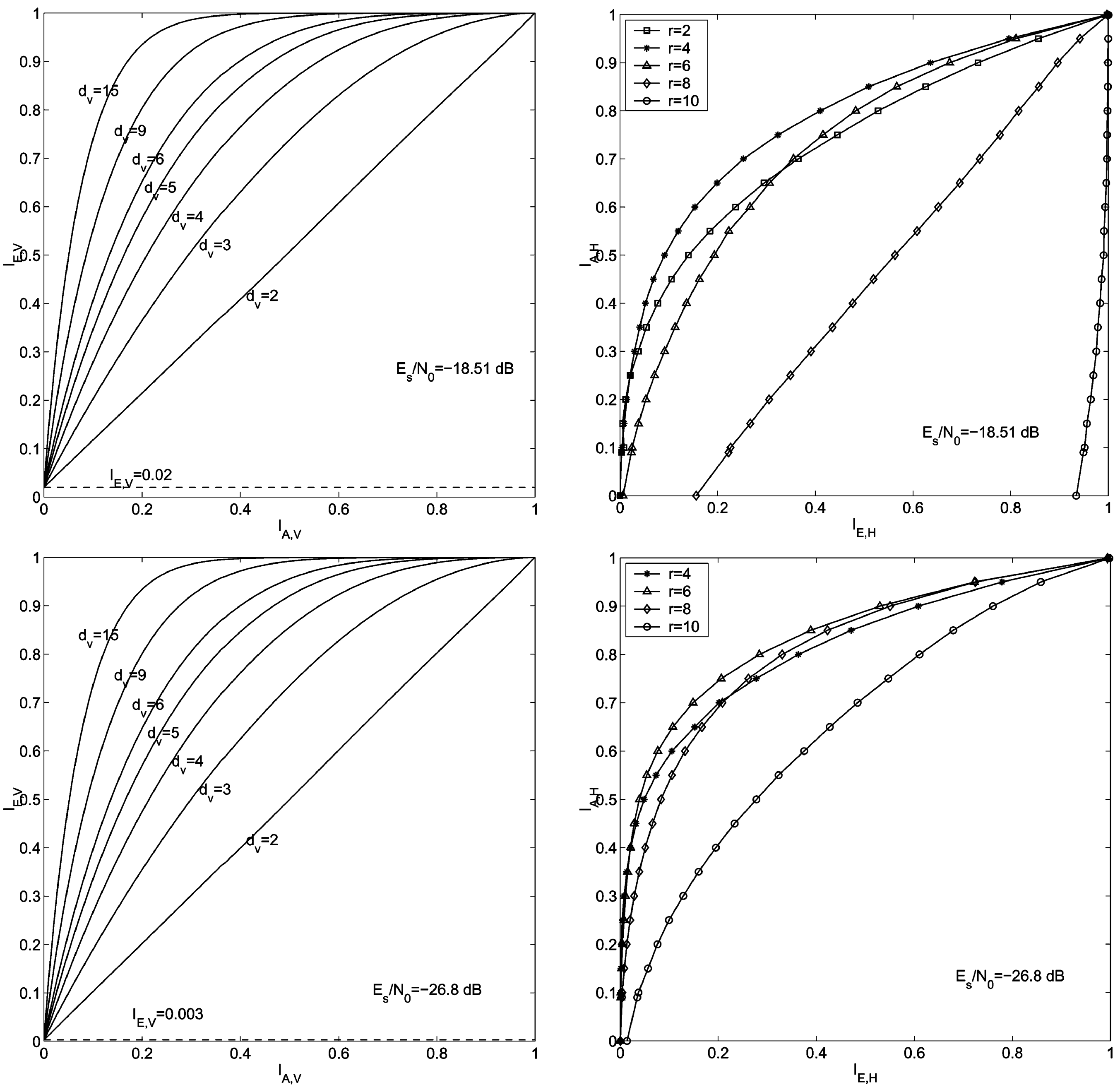


Fig. 7. EXIT charts for extrinsic input/output at the variable nodes (left) and Hadamard check nodes (right) with $E_s/N_0 = -18.51$ dB (up) and $E_s/N_0 = -26.8$ dB (bottom).

information $I_{E,H}$ evolves to one as the input increases to one. We also find that because degree-1 nodes attached at the super check nodes receive the information from the communication channel, there is an initial offset even when the input extrinsic information is zero. As shown in Appendix A, this initial offset is a function of $m_{\rm ch}$ and Hadamard order r. Next we evaluate the component EXIT curves of LDPC-Hadamard codes at another SNR, $E_s/N_0 = -26.8$ dB, as shown in the lower part of Fig. 7. The following observations are made in our experimental study.

- For a fixed E_s/N_0, the offset is higher for a larger r.
- For a certain r, when E_s/N_0 is small enough, the EXIT curves of the super check nodes are concave with small initial offsets.
- For a certain r, when E_s/N_0 increases, their initial offsets increase, and the EXIT curves of the super check nodes gradually change from concave to convex (except that the curve for $r = 2$ always remains concave). Such change is faster for a curve with a larger r.
- The EXIT curves of the variable nodes remain concave for all E_s/N_0 values.

Note that convexity and large offset of the EXIT curves for the super check nodes make matching difficult. Therefore we can conclude that the proposed LDPC-Hadamard codes are most suitable for designing good low-rate in the very-low SNR region. For higher rate codes working in higher SNR region where the EXIT curves of the LDPC-Hadamard codes with the smallest r still have a large offset, other alternatives (such as the standard LDPC codes) may offer better solutions.

A similar design strategy can be developed for nonsystematic Hadamard codes, but we will omit the details.

3) Code Rate: With the modified code graph, the LDPC-Hadamard codeword consists of all the variable nodes and degree-1 nodes attached at the check nodes. We now redefine the profiles of the code ensemble for the LDPC-Hadamard codes built on the modified code graph as follows. We only consider the proportions of the edges connected to the nondegree-1 nodes. Define λ_j as the fraction of the edges connected to the nodes of degree-j over all the edges connected to the nondegree-1 nodes and ρ_r as the fraction of edges that are connected to the check nodes. Once the profiles of $\{\lambda_j, \rho_r\}$ are fixed, the proportion of degree-1 nodes is determined. It is easily seen that the newly defined profiles are in fact the profiles of the LDPC precode.

The rates of LDPC-Hadamard codes built with systematic or nonsystematic Hadamard check nodes are then, respectively, given by

$$R_{\text{sys}} = 1 - \frac{\sum_{r=2,4,\ldots.} \frac{\rho_r(2^r-(r+1))}{(r+2)}}{\sum_{j=2}^{d_{v_{\max}}} \lambda_j/j + \sum_{r=2,4,\ldots.} \frac{\rho_r(2^r-(r+2))}{(r+2)}} \quad (45)$$

$$R_{\text{NS}} = 1 - \frac{\sum_{r=2,3,\ldots.} \frac{\rho_r(2^r-1)}{(r+2)}}{\sum_{j=2}^{d_{v_{\max}}} \lambda_j/j + \sum_r \frac{\rho_r(2^r-2)}{(r+2)}}. \quad (46)$$

For an LDPC-Hadamard code with nonsystematic check nodes, if we transmit the Hadamard coded bits attached at the check nodes including $\tilde{b}(j)$, we can avoid the transmission of the variable nodes corresponding to the information bits $b(j)$. However, it is possible that a variable node which connected to the position of $b(j)$ at one check node is also connected to the position of $\{c(0), c(2^r-1)\}$ at another check node while the nodes connected to positions of $\{c(0), c(2^r-1)\}$ should be transmitted. An ad hoc solution to solve this problem is to puncture the nondegree-1 nodes. The code rate for punctured nonsystematic LDPC-Hadamard codes is then given by

$$R_{\text{NS,Punc}} = \frac{\sum_j \lambda_j/j - \sum_r \rho_r/(r+2)}{\sum_j \lambda_j/j(1-P_0) + \sum_r \rho_r \frac{2^r-2}{r+2}} \quad (47)$$

where P_0 is the puncture rate. For a check node of order-r, there should be r of $r+2$ edges punctured. Therefore, a reasonable puncture rate for the nondegree-1 variable nodes is given by

$$P_0 = \sum_r \frac{\rho_r r}{r+2}. \quad (48)$$

For the sake of simplicity on the code design and analysis, we can also set $P_0 = 1$. More details on the derivation of code rates in (45) to (47) are described in Appendix C. Note that although we treat the LDPC-Hadamard codes with the systematic and nonsystematic check nodes separately, they can both exist in one LDPC-Hadamard code.

D. Code Optimization

Now we consider the optimization of the LDPC-Hadamard code ensemble for AWGN channels. We adopt a low-complexity optimization method based on approximating the density evolution as a one-dimensional dynamic system represented by an EXIT chart [14].

The EXIT chart of an LDPC-Hadamard code contains the average EXIT curves for the Hadamard check nodes, i.e., the functions $\mathcal{G}_V(I_{A,V}, \frac{E_s}{N_0})$ and $\mathcal{G}_H(I_{A,H}, \frac{E_s}{N_0})$ in (25) and (44), respectively. If there is no crossover between these two curves in the EXIT chart this ensures that the extrinsic mutual information $I_{E,V}$ can approach one. All the nondegree-1 nodes are then successfully decoded, indicating that all the information bits are also correctly decoded. We consider an example of a (d_v, r) LDPC-Hadamard code, i.e., in such code all the variable nodes are of degree-d_v, and all the check nodes are of order-r. The check nodes are systematic Hadamard codes. The EXIT curves are shown in Fig. 8. We set $E_s/N_0 = -18.51$ dB, i.e., $\sigma^2 = 35.48$. The EXIT curves for $d_v = 4$, $d_v = 5$, and $r = 6$ are illustrated. It can be seen that there is a crossover between the EXIT curves of $d_v = 4$ and $r = 6$ before approaching one, indicating that there exists a decoding error for the $(4, 6)$ LDPC-Hadamard codes at this SNR. However, there is no crossover between the EXIT curves of $d_v = 5$ and $r = 6$ before approaching one. Thus, the $(d_v = 5, r = 6)$ LDPC-Hadamard codes can asymptotically achieve the threshold of $E_s/N_0 = -18.51$ dB. Notice that this is the capacity limit of the code rate-0.02. The rate of the $(d_v = 5, r = 6)$ LDPC-Hadamard codes is $R = 0.01$. This means with the LDPC-Hadmard codes in a regular form, we can only achieve the limit of rate-0.02 by a lower rate code, indicating there is a rate loss for the regular LDPC-Hadamard code ensemble.

We next consider optimizing LDPC-Hadamard code ensembles with an irregular form. Code optimization can be achieved in principle by applying the standard density evolution based optimization procedure as in [6] to the modified LDPC-Hadamard code graph similarly to the design of LDPC codes. However, in practice this is quite difficult because of the extremely high computational complexity of the design procedure in the low-rate regime. It is also difficult to design the code ensemble with a good E_b/N_0 threshold from the density evolution with Gaussian approximation as in [5]. Here, we present a low-complexity code design method based on the EXIT chart.

In order to avoid tracking the evolution of EXIT curves during the optimization, we assume that the decoding input at the variable nodes is symmetric Gaussian with the same extrinsic mutual information as that from the output of the check node decoding. The same assumption is made for the EXIT curve of check nodes. With this assumption, we can evaluate the two curves in the EXIT chart separately and process the optimization without simulating the iterative decoding. The complexity of code ensemble optimization is then significantly reduced. Such a strong assumption has been proved to be valid for the LDPC optimization in many single link channels, as well as in ergodic multiple-input–multiple-output (MIMO) channels [12], [29]. We now utilize it for LDPC-Hadamard code optimization

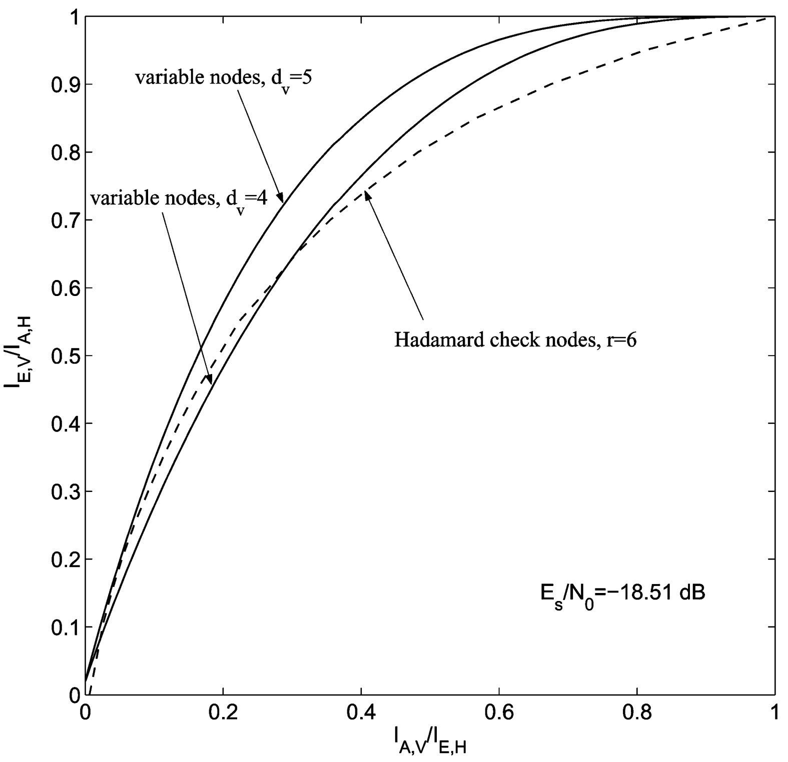


Fig. 8. EXIT charts for a ($\lambda_{d_v} = 1, \rho_r = 1$) LDPC-Hadamard code, $r = 6$.

with both nodes in the bipartite graph receiving information from the communication channels. We also assume that the Hadamard check nodes are regular, i.e., all the Hadamard check nodes have the same order. This assumption may limit the search space of the code ensemble, but further simplifies the code design.

Using the *convergence property* [14] and the *area property* [27] of the EXIT chart, we can form the design rules for LDPC-Hadamard codes. Given E_s/N_0, the LDPC-Hadamard code ensemble can be designed by enforcing the EXIT transfer function of the decoder to satisfy $\mathcal{G}_H^{-1}\left(x, \frac{E_s}{N_0}\right) < \mathcal{G}_V\left(x, \frac{E_s}{N_0}\right)$ and maximizing the code rate. It is easy to verify from (45)–(47) that with a fixed r, maximizing the code rate can be equivalently achieved by maximizing $\sum_j \lambda_j/j$. Hence, the optimization problem can be summarized as follows:

$$\begin{aligned} \underset{\{\lambda_j\}}{\text{maximize}} \quad & \sum_{j=2}^{d_{v_{\max}}} \lambda_j/j \\ \text{s.t.} \quad & \mathcal{G}_V\left(x, \frac{E_s}{N_0}\right) > \mathcal{G}_H^{-1}\left(x, \frac{E_s}{N_0}\right) \\ & \text{with} \sum_{j=2}^{d_{v_{\max}}} \lambda_j = 1, \quad 0 \le x < 1. \end{aligned} \tag{49}$$

Given the capacity limit E_s/N_0 of a certain code rate, we can obtain $\mathcal{G}_H^{-1}(x, m_{\text{ch}})$ using Monte Carlo simulations. We then obtain the optimized $\{\lambda_j\}$ by solving the above optimization problem, which can be easily solved using linear programming [35]. Two optimization results are shown in Fig. 9. The upper one is for the systematic codes at the check nodes with $\rho_4 = 1$. The designed rate is $R = 0.05$ and the designed threshold is -1.35 dB. The lower one is for the nonsystematic codes with $\rho_5 = 1$. In this case, we set the puncture rate $P_0 = 1$. The code rate is $R = 0.022$ and the threshold from EXIT chart optimization is -1.50 dB. It can be seen that in both cases, the optimized EXIT curve of function $\mathcal{G}_V\left(x, \frac{E_s}{N_0}\right)$ matches very well with that of function $\mathcal{G}_H^{-1}\left(x, \frac{E_s}{N_0}\right)$. The resulting curve of $\mathcal{G}_H^{-1}\left(x, \frac{E_s}{N_0}\right)$ is always below that of function $\mathcal{G}_V\left(x, \frac{E_s}{N_0}\right)$. The performance of designed codes will be evaluated in Section V.

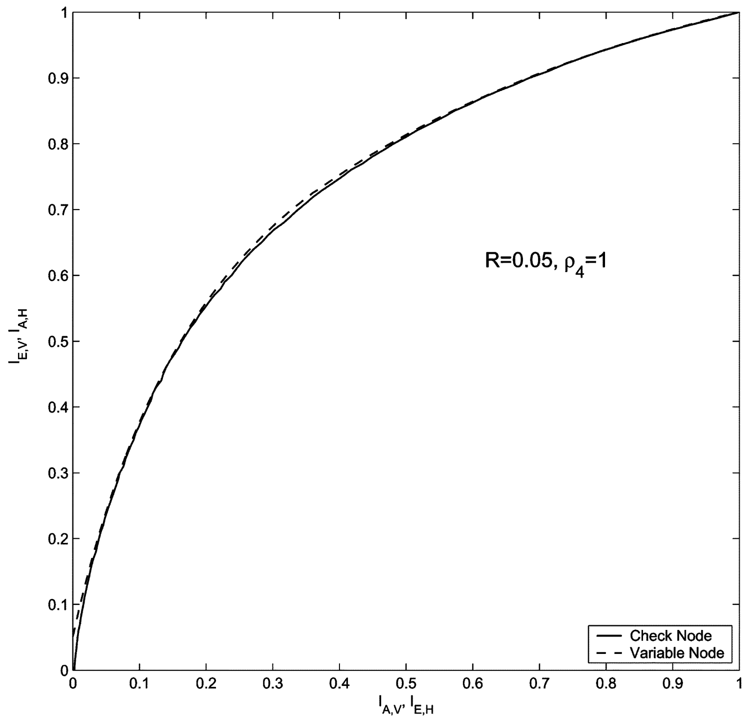
V. Simulation Results

In this section, we present performance results of the optimized LDPC-Hadamard codes from computer simulations.

A. Approaching the Low-Rate Limit

We consider several examples to demonstrate that the optimized LDPC-Hadamard codes exhibit performance approaching the low-rate Shannon limit.

Example of an LDPC-Hadamard code with $r = 4$: In the first example, we consider the LDPC-Hadamard code with systematic Hadamard check nodes of order $r = 4$, i.e., $\rho_4 = 1$. The designed LDPC-Hadamard code has the rate $R = 0.050$. The profile of the variable nodes from the edge perspective is given by $\{\lambda_2 = 0.2377, \lambda_3 = 0.2671, \lambda_8 = 0.1469, \lambda_9 = 0.3173, \lambda_{21} = 0.0310\}$. The threshold from EXIT chart optimization is $-$ 1.35 dB. The BER performance is illustrated in Fig. 10. The BER is measured by counting errors on nondegree-1 variable nodes. We apply the same way to measure the BER for all other examples in this section. The performance of LDPC codes with $R = 0.05$ and low-rate turbo Hadamard codes [18] with various orders is also illustrated in the same figure for comparison. The degree distributions of the rate-0.05 LDPC



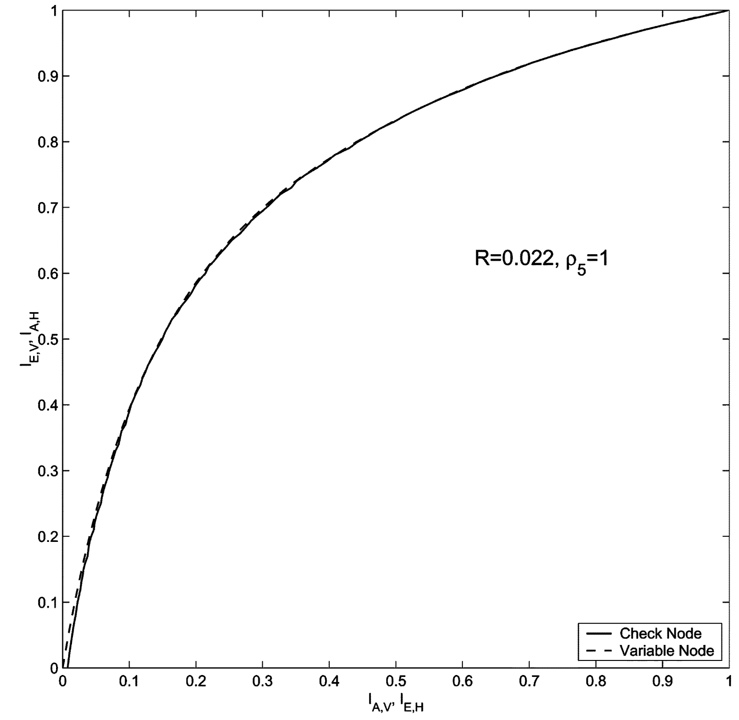


Fig. 9. EXIT charts for optimized LDPC-Hadamard codes.

code are $\{\lambda_2 = 0.5466, \lambda_3 = 0.1708, \lambda_4 = 0.0267, \lambda_5 = 0.0219, \lambda_6 = 0.1005, \lambda_7 = 0.0259, \lambda_{13} = 0.0448, \lambda_{18} = 0.0321, \lambda_{20} = 0.0077, \lambda_{21} = 0.0119, \lambda_{22} = 0.0111\}$ and $\{\rho_2 = 0.1, \rho_3 = 0.9\}$, obtained from [30] by setting maximum left degree 30. The number of concatenated component codes in the turbo Hadamard codes is $M = 3$. The rates of turbo Hadamard codes with $r = 5$ and 6 are 0.058 and 0.033, respectively. For a fair comparison, we set the information length for all these codes to be approximately equal, i.e., $K \approx 65536$. The numbers of iterations for message-passing decoding of LDPC-Hadamard codes and LDPC codes are set to be 400 and 2000, respectively. The number of iterations we set for THC is 50. The results show that the LDPC-Hadamard code has a 10^{-5} BER performance at $E_b/N_0 = -1.18$ dB, outperforming the turbo Hadamard code with the closest rate ($r = 5, R = 0.058$) by 0.48 dB and LDPC codes with the same rate by 1.67 dB. It is seen that the designed LDPC-Hadamard code even performs 0.23 dB better than the turbo Hadamard code ($r = 6$, $R = 0.033$) with lower rate. This threshold from the simulation is only 0.17 dB from the designed threshold. Moreover, it is only 0.26 dB shy

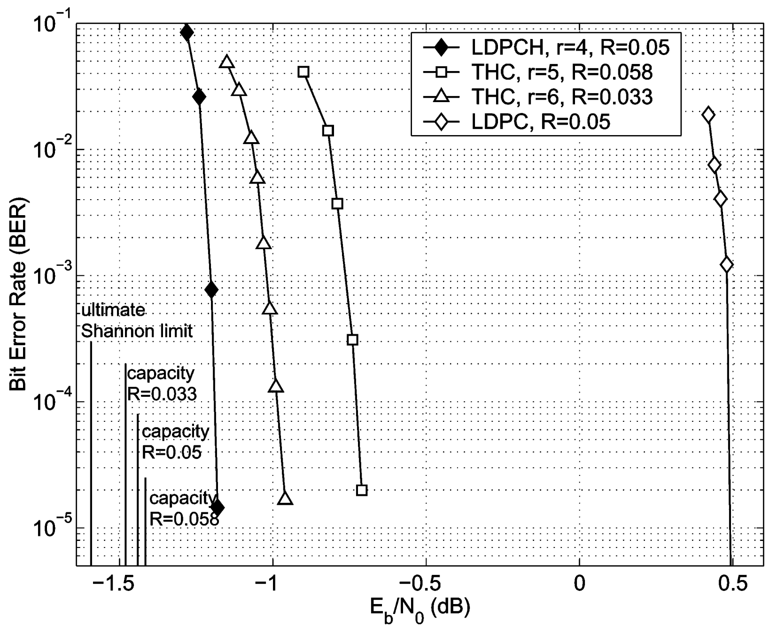


Fig. 10. BER performance of designed LDPC-Hadamard code with $r = 4$ and turbo Hadamard codes with $M = 3$, $r = 5, 6$. $K \approx 65536$.

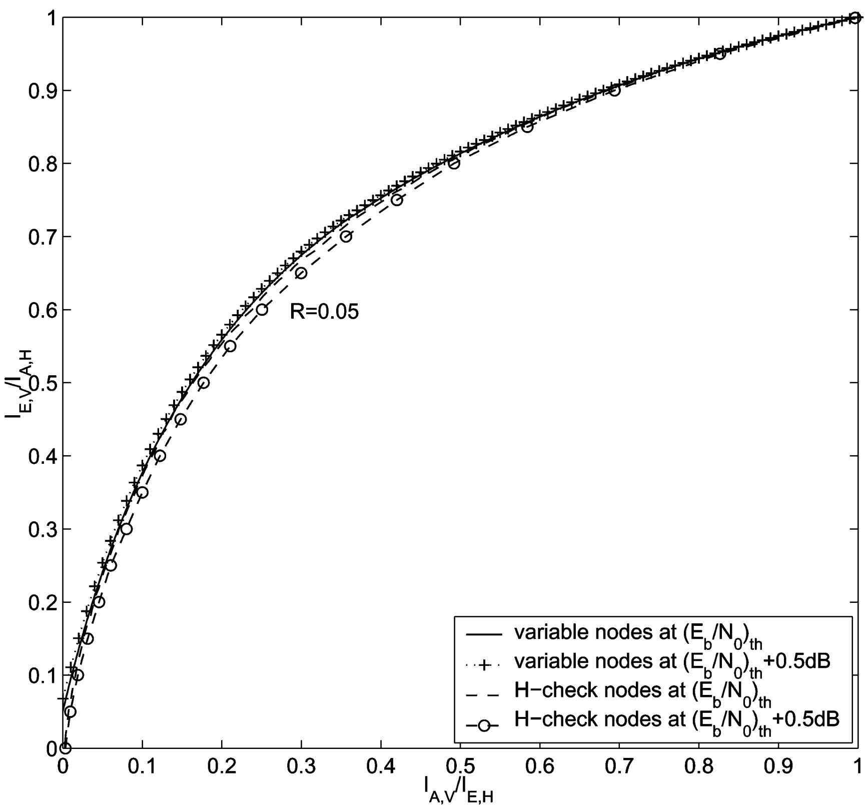


Fig. 11. EXIT curves of variable nodes and check nodes of a rate-0.05 LDPC-Hadamard code ensemble.

of the Shannon capacity for $R = 0.05$ with binary input (-1.44 dB) and 0.41 dB away from the ultimate low-rate limit. Notice that in this example, the order of the Hadamard check nodes is $r = 4$, which is less than that of the turbo Hadamard codes shown in the same figure; thus, the decoding complexity is much lower.

We can see that the simulation result of the LDPC-Hadamard code matches very well with the designed threshold. To explain this, we use the EXIT chart analysis similarly as before. The EXIT curves of the variable nodes and check nodes of above optimized LDPC-Hadamard code ensemble are illustrated in Fig. 11. Similar to the discussion for Fig. 2, we examine two

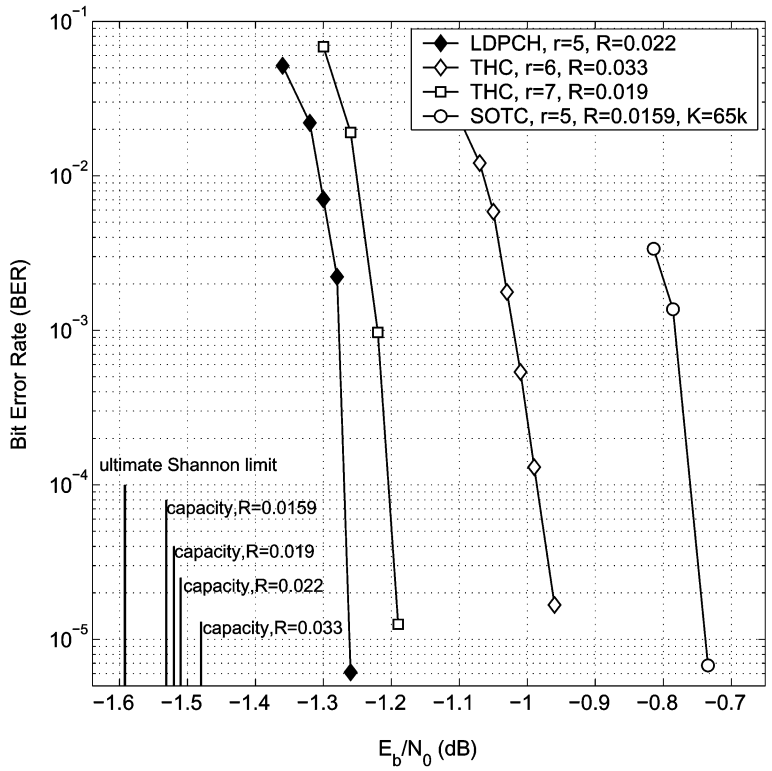


Fig. 12. BER performance of designed LDPC-Hadamard code with $r = 5$, turbo Hadamard codes with $M = 3$, $r = 6, 7$, and the SOTC with $M = 2$, $r = 5$. $K \approx 65536$.

SNR values, i.e., $(E_b/N_0)_{th}$ and $(E_b/N_0)_{th}$+0.5 dB. Since the Hadamard check nodes receive the information from the communication channel, the EXIT curve of Hadamard check nodes also changes with the SNR increase. From Fig. 11, it is seen that though there is little change for the variable curve, a more noticeable change takes place for the Hadamard check curve with the SNR increase. A wide iterative decoding tunnel is obtained by increasing SNR, which indicates that the performance of LDPC-Hadamard codes with finite block length from the simulations will approach the optimized threshold. Compared with Fig. 2, we can clearly see the advantage of the proposed LDPC-Hadamard code over the conventional LDPC code at low rate.

Example of an LDPC-Hadamard code with $r = 5$: We now consider an LDPC-Hadamard code with the check nodes having an odd order, $r = 5$. In this case, according to Theorem 4.2, the check nodes have to be the nonsystematic Hadamard codes. We set the puncture rate $P_0 = 1$. The profile of the optimized LDPC-Hadamard code ensemble is $\{\lambda_2 = 0.2096, \lambda_3 = 0.1700, \lambda_5 = 0.1435, \lambda_6 = 0.1137, \lambda_{13} = 0.3632, \rho_5 = 1\}$. The rate $R = 0.022$ and the designed E_b/N_0 threshold from EXIT chart is $-$ 1.50 dB. The BER performance of the LDPC-Hadamard code, the turbo Hadamard codes with $r = 6$, $R = 0.033$ and $r = 7$, $R = 0.019$, and the superorthogonal turbo codes (SOTC) [36] is shown in Fig. 12. We set $M = 2$, $r = 5$ for the superorthogonal turbo codes. The rate is then 0.0159. It can be seen that the E_b/N_0 threshold at 10^{-5} BER of the LDPC-Hadamard code is -1.26 dB, 0.24 dB from designed threshold. It performs 0.52 and 0.3 dB better than that of the SOTC code and the turbo Hadamard code with $r = 6$, respectively. It performs 0.08 dB better than the best turbo Hadamard code with $r = 7$. The gap from the capacity threshold for rate-0.022 codes is only 0.25 dB.

Examples of LDPC-Hadamard codes with high orders: We now consider the design of LDPC-Hadamard codes with higher orders and lower rates to push the performance closer to the ultimate Shannon limit. Codes with $r = 8$, $R = 0.0080$, and $r = 10$, $R = 0.0030$, are designed and the resulting optimized code ensemble profiles are $\{\lambda_2 = 0.2268, \lambda_3 = 0.2252, \lambda_4 = 0.3918, \lambda_{12} = 0.0400, \lambda_{13} = 0.1162, \rho_8 = 1\}$ and $\{\lambda_2 = 0.2608, \lambda_3 = 0.5778, \lambda_{15} = 0.0877, \lambda_{16} = 0.0737, \rho_{10} = 1\}$ with the designed thresholds from EXIT charts are -1.53 dB and -1.55 dB, respectively. Fig. 13 illustrates the BER performance of these LDPC-Hadamard codes with various code lengths, as well as the best performance that low-rate turbo Hadamard codes can achieve. It is seen that the E_b/N_0 threshold of the ($r = 8$, $R = 0.008$) LDPC-Hadamard code with $K = 71\,000$ is -1.36 dB, 0.17 dB from the designed threshold and improving the best threshold of the turbo Hadamard, $E_b/N_0 = -1.18$ dB, by 0.18 dB. The gap between performance of this LDPC-Hadamard code and the ultimate low-rate limit is now only 0.23 dB. By increasing the code length to $K = 238\,000$, the threshold is improved to $-$ 1.38 dB, and the gap to the Shannon limit is reduced to 0.21 dB. As shown in Fig. 13, with an extremely low-rate approach, i.e., $r = 10$, $R = 0.0030$, and extremely long code, i.e., $K = 650\,000$, the E_b/N_0 threshold from the simulation is -1.44 dB, 0.11 dB from the designed threshold, which is only 0.15 dB from the theoretical Shannon limit for the code rate approaching zero. To the best of our knowledge, the performance reported in Fig. 13 is the best among those reported in the literature.

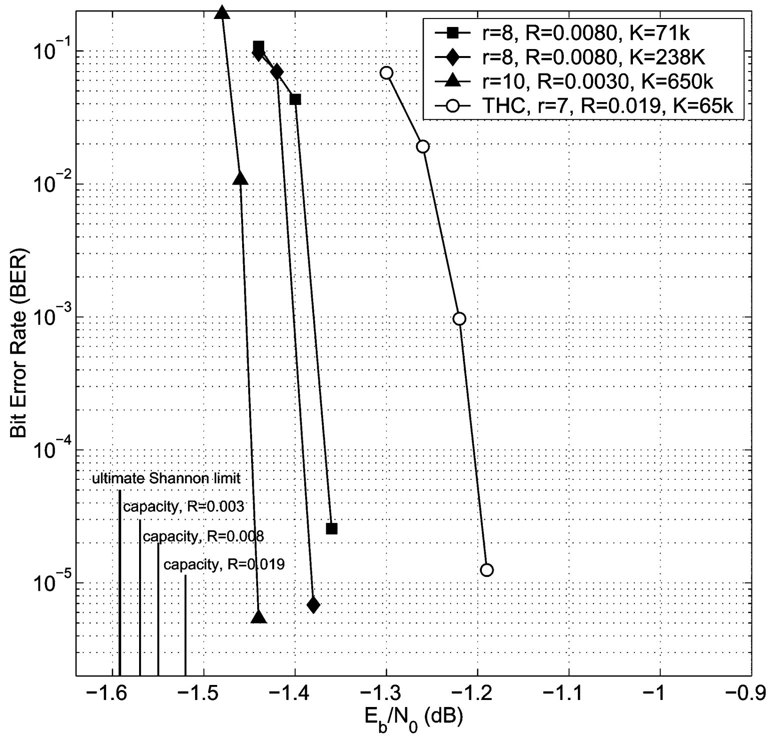


Fig. 13. BER performance of designed LDPC-Hadamard codes and turbo Hadamard codes with $M = 3$ and $r = 7$.

It is known that Hamming and BCH codes [11], [24] can be used to construct the GLDPC codes. However, as discussed earlier, there are some difficulties about GLDPC codes, such as efficient APP decoding techniques, the EXIT mismatching problem and the existence of degree-1 nodes. We have resolved these difficulties for the proposed LDPC-Hadamard codes by exploiting some nice properties of Hadamard codes, such as the efficient APP-FHT decoding algorithm and the underlying SPC constraints shown in Theorem 4.2 and Appendix B. Similar or different solutions may also be possible for other block codes, but more research work is required on this issue.

B. Convergence Rate and Decoding Complexity

We now investigate the convergence rate of decoding LDPC-Hadamard codes. We consider an LDPC-Hadamard code with rate $R = 0.05$ and $K \approx 65536$. The code ensemble is the same as that in the first example in Section V-A.

Fig. 14 illustrates the evolution of the bit-error rate as a function of the iteration number for $E_b/N_0 = -1.18$, -1.15 dB, -1.00 dB, and 0 dB. The results are the averages from 20 simulations. We can see that it takes 217 iterations for LDPC-Hadamard decoding to converge for $E_b/N_0 = -1.18$ dB. Note that there is a gap of 0.22 dB between the operating point of $E_b/N_0 = -1.18$ dB and the capacity of $E_b/N_0 = -1.44$ dB for code rate of 0.05. The convergence rate of 217 iterations for the LDPC-Hadamard code is quite similar to a standard LDPC code of rate-$\frac{1}{2}$ operating at E_b/N_0 value at about 0.22 dB from the corresponding capacity of 0.2 dB. This indicates that the iteration number required for both standard LDPC and LDPC-Hadamard codes are determined by their relative distances toward the individual capacity values.

On the other hand, we may be interested in the convergence rate for a fixed E_b/N_0 value for different codes. (Note that this may not be a fair comparison, since LDPC-Hadamard codes usually have lower rates and so require more redundancy. However, such a comparison may be useful when power efficiency and decoding complexity are primary concerns and spectrum efficiency is less an issue, such as for UWB and deep space applications.) From Fig. 14, we can see that with E_b/N_0 increased to 0 dB, the number of iterations to converge dramatically reduces to only 38. This observation is very interesting. Since for a standard LDPC code, many more iterations are required to operate at $E_b/N_0 = 0$ dB. (We have built a rate 0.33 LDPC code using the ensemble from [30]. We simulated this code for information length $K = 65536$. We observed that at $E_b/N_0 = 0$ dB, it took 95 iterations to converge.)

For extremely low-rate LDPC-Hadamard codes with high orders, e.g., $r = 8$ or 10, we can further reduce the decoding complexity. Because of the special structure of the LDPC-Hadamard code in Fig. 5, among 2^r edges in each Hadamard code, only $r+2$ extrinsic messages are updated during each decoding iteration. Others are all degree-1 nodes and their outbound extrinsic messages remain unchanged during the whole decoding process. Therefore, the overall complexity of decoding LDPC-Hadamard codes can be reduced if we only compute $L_{ext}(i)$ for the edges in the set $\mathcal{S}$, where $\mathcal{S}_r = \{b(0), b(1), \ldots, b(r), c(2^r - 1)\}$. This can be achieved via reduced APP fast Hadamard transforms. The number of additions in APP-FHT is reduced from $O(r2^r)$ to $O(3 \times 2^r)$. The details are provided in Appendix D.

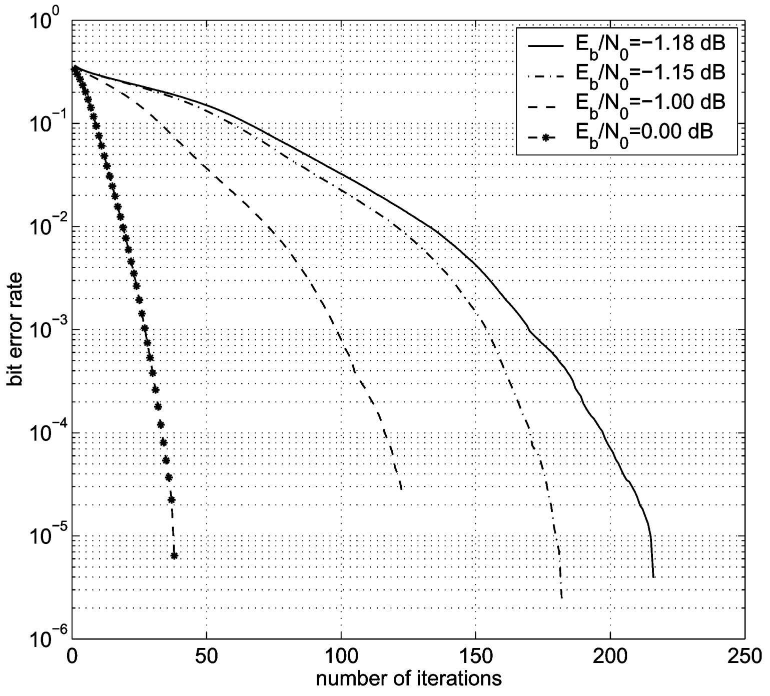


Fig. 14. Decoding convergence of LDPC-Hadamard illustrated by BER evolution as a function of the iteration number $K \approx 65536, R = 0.05$.

VI. Conclusion

In this paper, we have introduced generalized low-density parity-check (GLDPC) codes. In particular, we have studied GLDPC codes with Hadamard constraints, referred to as LDPC-Hadamard codes. We have presented a message-passing-based low-complexity soft-input soft-output (SISO) decoding algorithm for LDPC-hadamard codes that employs fast APP-FHT decoding for Hadamard check nodes. The performance of the GLDPC codes are discussed, based on which a modified LDPC-Hadamard code graph is proposed in order to obtain performance approaching the Shannon limit. We also presented a low-complexity optimization method for LDPC-Hadamard code ensembles based on the EXIT chart characteristics. Simulation results show that optimized LDPC-Hadamard codes perform better than low-rate turbo Hadamard codes while also having a fast convergence rate. A rate-0.003 LDPC-Hadamard code with long block length has performance only 0.15 dB away from the ultimate Shannon limit and 0.24 dB better than the best low-rate turbo-Hadamard codes.

Appendix A
APP Decoding of Hadamard Code

Define $\tilde{\boldsymbol{L}}_{\rm apr}(i) \triangleq [L_{\rm apr}(0), \ldots, L_{\rm apr}(i-1), 0, L_{\rm apr}(i+1), \ldots, L_{\rm apr}(2^r-1)]^T$. From (21), $L_{\rm app}(i)$ can then be written as shown in (A1) at the bottom of the page.

Now consider decoding of Hadamard check nodes described in Section IV-C. Define $\tilde{\boldsymbol{L}}^1_{apr}(i)$ as a length-2^r vector with nonzero entries $L_{apr}(i')$ for $i' = 0, 1, 2, 4, \ldots, 2^{r-1}, 2^r - 1$, $i' \neq i$, and zero entries elsewhere. Similarly, define $\tilde{\boldsymbol{L}}^2_{\rm apr}(i)$ with nonzero entries $L_{\rm apr}(i')$ except that $i' =$

$$\begin{aligned} L_{\rm app}(i) &= \log \frac{\sum_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\pm 1\}} \exp\left(\frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \frac{2\boldsymbol{x}}{\sigma^2}\right\rangle + \frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \tilde{\boldsymbol{L}}_{\rm apr}(i)\right\rangle\right) \exp\left(\frac{1}{2}L_{\rm apr}(i)\right)}{\sum_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\mp 1\}} \exp\left(\frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \frac{2\boldsymbol{x}}{\sigma^2}\right\rangle + \frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \tilde{\boldsymbol{L}}_{\rm apr}(i)\right\rangle\right) \exp\left(-\frac{1}{2}L_{\rm apr}(i)\right)} \\ &= \underbrace{\log \frac{\sum_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\pm 1\}} \exp\left(\frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \frac{2\boldsymbol{x}}{\sigma^2}\right\rangle + \frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \tilde{\boldsymbol{L}}_{\rm apr}(i)\right\rangle\right)}{\sum_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\mp 1\}} \exp\left(\frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \frac{2\boldsymbol{x}}{\sigma^2}\right\rangle + \frac{1}{2}\left\langle \pm \boldsymbol{h}^j, \tilde{\boldsymbol{L}}_{\rm apr}(i)\right\rangle\right)}}_{L_{\rm ext}(i)} + \log \frac{\exp\left(\frac{1}{2}L_{\rm apr}(i)\right)}{\exp\left(-\frac{1}{2}L_{\rm apr}(i)\right)} \\ &= L_{\rm ext}(i) + L_{\rm apr}(i). \end{aligned} \tag{A1}$$

$$\begin{aligned}L_{\text{ext}}(i) &= \log \frac{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\pm 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^1_{\text{apr}}(i)\right\rangle + \frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^2_{\text{apr}}(i)\right\rangle\right)}{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\mp 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^1_{\text{apr}}(i)\right\rangle + \frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^2_{\text{apr}}(i)\right\rangle\right)} \\ &= \log \frac{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\pm 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^1_{\text{apr}}(i)\right\rangle \pm a_j\right)}{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\mp 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^1_{\text{apr}}(i)\right\rangle \pm a_j\right)} \\ &= \log \frac{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\pm 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^1_{\text{apr}}(i) + b_j\tilde{\mathbf{1}}\right\rangle\right)}{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\mp 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^1_{\text{apr}}(i) + b_j\tilde{\mathbf{1}}\right\rangle\right)}\end{aligned} \tag{A3}$$

$0, 1, 2, 4, \ldots, 2^{r-1}, 2^r - 1$. Obviously, $\tilde{\boldsymbol{L}}^1_{\text{apr}}(i)$ are extrinsic inputs of SPC part and $\tilde{\boldsymbol{L}}^2_{\text{apr}}(i)$ are extrinsic inputs from degree-1 nodes. We then have

$$\tilde{\boldsymbol{L}}_{\text{apr}}(i) = \tilde{\boldsymbol{L}}^1_{\text{apr}}(i) + \tilde{\boldsymbol{L}}^2_{\text{apr}}(i). \tag{A2}$$

For the check nodes, $\frac{2\boldsymbol{x}}{\sigma^2} = \mathbf{0}$. Also the extrinsic input $\tilde{\boldsymbol{L}}^2_{\text{apr}}(i)$ from the degree-1 nodes is a constant vector. With some manipulations, the extrinsic output $L_{ext}(i)$ can be written as shown in (A3) at the top of the page, where a_j and b_j are some variables only depends on $\tilde{\boldsymbol{L}}^2_{\text{apr}}(i)$ and $\boldsymbol{h}_j$, $\tilde{\mathbf{1}}$ is a vector with unit entries for positions $i' = 0, 1, 2, 4, \ldots, 2^{r-1}, 2^r - 1$, and $i' \neq i$. It is seen that for $i = 0, 1, 2, 4, \ldots, 2^{r-1}, 2^r - 1$, above equation is exact for computing extrinsic output of SPC. When $\tilde{\boldsymbol{L}}^1_{apr}(i) = \mathbf{0}$, indicating extrinsic input is zero, we have

$$L_{\text{ext}}(i) = \log \frac{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\pm 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^2_{\text{apr}}(i)\right\rangle\right)}{\sum\limits_{\boldsymbol{h}^j \in \{\boldsymbol{H}: H(i,j)=\mp 1\}} \exp\left(\frac{1}{2}\left\langle \pm\boldsymbol{h}^j, \tilde{\boldsymbol{L}}^2_{\text{apr}}(i)\right\rangle\right)}. \tag{A4}$$

We know $\tilde{\boldsymbol{L}}^2_{apr}(i)$ are actually channel outputs, thus the pdf of each nonzero entry only depends on E_s/N_0 or m_{ch}. Then the initial offset of the EXIT curve for Hadamard check nodes is only a function of m_{ch}, and the number of nonzero entries in $\tilde{\boldsymbol{L}}^2_{apr}(i)$, in other words, the Hadamard order r.

APPENDIX B
PROOF OF THEOREM 4.2

Proof: Theorem 4.2 is proved by induction. First, consider a systematic Hadamard code. We have $b(j) = c(2^{j-1})$, $j = 1, \ldots, r$ and $b(0) = c(0)$. Hence, (43) can be written as

$$c(2^r - 1) \oplus c(0) \oplus \sum_{j=1}^{r}\!\oplus\, c(2^{j-1}) = 0. \tag{B1}$$

It is easy to verify that (B1) is satisfied by a $r = 2$ systematic code since it is an SPC code. Now we assume (B1) is also true for $r = m$, i.e.

$$\begin{aligned}&c(2^m - 1) \oplus c(0) \oplus \sum_{j=1}^{m}\!\oplus\, c(2^{j-1}) = 0 \\ &\Rightarrow c(2^m - 1) = c(0) \oplus \sum_{j=1}^{m}\!\oplus\, c(2^{i-1}).\end{aligned} \tag{B2}$$

Then, the Hadamard matrix in binary form with order $r = m+2$ is given by

$$\boldsymbol{H}_{4n} = \begin{bmatrix} \boldsymbol{H}_n & \boldsymbol{H}_n & \boldsymbol{H}_n & \boldsymbol{H}_n \\ \boldsymbol{H}_n & \overline{\boldsymbol{H}}_n & \boldsymbol{H}_n & \overline{\boldsymbol{H}}_n \\ \boldsymbol{H}_n & \boldsymbol{H}_n & \overline{\boldsymbol{H}}_n & \overline{\boldsymbol{H}}_n \\ \boldsymbol{H}_n & \overline{\boldsymbol{H}}_n & \overline{\boldsymbol{H}}_n & \boldsymbol{H}_n \end{bmatrix} \tag{B3}$$

where $n = 2^m$ and $\overline{\boldsymbol{H}}$ is denotes the binary complementary matrix of $\boldsymbol{H}$. We can show in $\boldsymbol{H}_{4n}$ that $c(2^{m+2} - 1)$ and $c(2^m - 1)$ are in the same position as in maxtrix $\boldsymbol{H}_n$ for any column, similarly for $c(2^m)$ and $c(2^{m+1})$. From (B3), for all the columns or the codeword obtained from $\boldsymbol{H}_{4n}$, we have

$$c(2^{m+2} - 1) \oplus c(2^m - 1) = c(2^m) \oplus c(2^{m+1}). \tag{B4}$$

Then, we have

$$\begin{aligned}&c(2^{m+2} - 1) \oplus c(2^m - 1) \oplus c(2^m) \oplus c(2^{m+1}) = 0 \\ &\Rightarrow c(2^{m+2} - 1) \oplus c(0) \oplus \sum_{j=1}^{m+2}\!\oplus\, c(2^{j-1}) = 0.\end{aligned} \tag{B5}$$

Then, it is satisfied for $\overline{\boldsymbol{H}}_{4n}$. For odd r, we can simply check a special case of $r = 3$ and find that it is not satisfied for $\overline{\boldsymbol{H}}_8$. With similar induction procedures, we can prove (B1) is not satisfied for all odd r.

Now we consider the nonsystematic Hadamard codes. From (8), we have $c(2^{j-1}) = \tilde{b}(j) = b(0) \oplus b(j)$ for $j = 1, \ldots, r$. Then $b(j) = b(0) \oplus c(2^{j-1})$. Since $b(0) = c(0)$, (43) can then be written as

$$c(2^r - 1) \oplus \sum_{j=0}^{r}\!\oplus\, c(0) \oplus \sum_{j=1}^{r}\!\oplus\, c(2^{j-1}) = 0. \tag{B6}$$

The above equation is easily verified for $r=2$. Assume (B6) is true for $r=m$, i.e.

$$c(2^m-1)\oplus \sum_{j=0}^{m}\!\oplus\, c(0)\oplus \sum_{j=1}^{m}\!\oplus\, c(2^{i-1})=0$$

$$\Rightarrow c(2^m-1)=\sum_{j=0}^{m}\!\oplus\, c(0)\oplus \sum_{j=1}^{m}\!\oplus\, c(2^{j-1}). \qquad \text{(B7)}$$

Now consider $r=m+1$. We have

$$\boldsymbol{H}_{2n}=\begin{bmatrix}\boldsymbol{H}_n & \boldsymbol{H}_n\\ \boldsymbol{H}_n & \overline{\boldsymbol{H}}_n\end{bmatrix},\quad \overline{\boldsymbol{H}}_{2n}=\begin{bmatrix}\overline{\boldsymbol{H}}_n & \overline{\boldsymbol{H}}_n\\ \overline{\boldsymbol{H}}_n & \boldsymbol{H}_n\end{bmatrix}. \qquad \text{(B8)}$$

For every column in $\boldsymbol{H}_{2n}$ and $\overline{\boldsymbol{H}}_{2n}$, we have

$$c(2^{m+1}-1)\oplus c(2^m-1)=c(2^m)\oplus c(0). \qquad \text{(B9)}$$

Then, we have

$$\begin{aligned}&c(2^{m+1}-1)\oplus c(2^m-1)\oplus c(2^m)\oplus c(0)=0\\ &\Rightarrow c(2^{m+1}-1)\oplus \sum_{j=0}^{m}\!\oplus\, c(0)\oplus \sum_{j=1}^{m}\!\oplus\, c(2^{j-1})\\ &\quad\oplus c(2^m)\oplus c(0)=0\\ &\Rightarrow c(2^{m+1}-1)\oplus \sum_{j=0}^{m+1}\!\oplus\, c(0)\oplus \sum_{j=1}^{m+1}\!\oplus\, c(2^{j-1})=0.\end{aligned} \qquad \text{(B10)}$$

□

APPENDIX C
CODE RATE OF LDPC-HADAMARD CODES

Assume that the total number of edges is n, excluding the edges connected to the degree-1 nodes. The total number of check nodes, the total number of constraints, and the number of the nondegree-1 nodes is then $\sum_r \frac{n\rho_r}{r+2}$, $\sum_r \frac{n\rho_r(2^r-(r+1))}{r+2}$, and $\sum_i \frac{n\lambda_i}{i}$, respectively. For an LDPC-Hadamard code with systematic Hadamard check nodes of even order r, the number of the degree-1 nodes is $\sum_r \frac{n\rho_r}{r+2}(2^r-(r+2))$. The code rate is then given by

$$\begin{aligned}R_{\text{sys}}&=1-\frac{\sum_r \frac{n\rho_r(2^r-(r+1))}{(r+2)}}{\sum_j n\lambda_j/j+\sum_r \frac{n\rho_r(2^r-(r+2))}{(r+2)}}\\ &=1-\frac{\sum_r \frac{\rho_r(2^r-(r+1))}{(r+2)}}{\sum_j \lambda_j/j+\sum_r \frac{\rho_r(2^r-(r+2))}{(r+2)}}.\end{aligned} \qquad \text{(C1)}$$

We can also obtain the code rate of the LDPC-Hadamard code from the corresponding LDPC pre-code. Denote by R^L, N_L, and N_H the code rate of the LDPC, length of the LDPC, and the length of the LDPC-Hadamard code, respectively. We have

$$R^L=1-\frac{\sum_r \rho_r/(r+2)}{\sum_j \lambda_j/j},\qquad \text{with}\ \sum_{i=2}^{d_{v\max}}\lambda_j=1 \qquad \text{(C2)}$$

$$N_L=\sum_j n\lambda_j/j, \qquad \text{(C3)}$$

$$N_H=\sum_j n\lambda_j/j+\sum_r \frac{n\rho_r(2^r-(r+2))}{r+2}. \qquad \text{(C4)}$$

The code rate of the LDPC-Hadamard code is then given by

$$\begin{aligned}R_{\text{sys}}&=R^L\frac{N_L}{N_H}=R^L\cdot\frac{\sum_j\lambda_j/j}{\sum_j\lambda_j/j+\sum_r\rho_r\frac{2^r-(r+2)}{r+2}}\\ &=\frac{\sum_j\lambda_j/j-\sum_r\frac{\rho_r}{r+2}}{\sum_j\lambda_j/j+\sum_r\rho_r\frac{(2^r-(r+2))}{(r+2)}}\\ &=\frac{\sum_j\lambda_j/j+\sum_r\frac{\rho_r(2^r-(r+2))}{r+2}-\sum_r\frac{\rho_r(2^r-(r+1))}{r+2}}{\sum_j\lambda_j/j+\sum_r\frac{\rho_r(2^r-(r+2))}{r+2}}\\ &=1-\frac{\sum_r\frac{\rho_r(2^r-(r+1))}{r+2}}{\sum_j\lambda_j/j+\sum_r\frac{\rho_r(2^r-(r+2))}{r+2}}.\end{aligned} \qquad \text{(C5)}$$

In an LDPC-Hadamard code with nonsystematic check nodes, the total number of constraints is now $\sum_r \frac{n\rho_r(2^r-1)}{r+2}$ since there are an additional r SPC constraints for $\{b(0),b(j),\tilde{b}(j)\}$. The number of degree-1 nodes is $N_L\sum_r\frac{n\rho_r(2^r-2)}{r+2}$. Similarly, we obtain the code rate, given by

$$R_{\text{NS}}=1-\frac{\sum_r\frac{\rho_r(2^r-1)}{r+2}}{\sum_j\lambda_j/j+\sum_r\frac{\rho_r(2^r-2)}{r+2}}. \qquad \text{(C6)}$$

We can also obtain the code rate through the expansion of the LDPC pre-code. Given the LDPC pre-code with rate R^L in (C4), the code rate of the LDPC-Hadamard code is given by

$$\begin{aligned}R_{\text{NS}}&=R^L\frac{\sum_j\lambda_j/j}{\sum_j\lambda_j/j+\sum_r\frac{\rho_r(2^r-2)}{r+2}}\\ &=\frac{\sum_j\lambda_j/j-\sum_r\frac{\rho_r}{r+2}}{\sum_i\lambda_j/j+\sum_r\frac{\rho_r(2^r-2)}{r+2}}\\ &=1-\frac{\sum_r\frac{\rho_r(2^r-1)}{r+2}}{\sum_j\lambda_j/j+\sum_r\frac{\rho_r(2^r-2)}{r+2}}.\end{aligned} \qquad \text{(C7)}$$

Now consider the punctured LDPC-Hadamard with nonsystematic check nodes. Given the puncture rate of the nondegree-1 nodes P_0, assuming the puncture rate is the same for any degree-j nodes, the length of coded bits is $N_H=\sum_j\lambda_j/j(1-P_0)+\sum_r\rho_r\frac{2^r-2}{r+2}$. The code rate is then given by

$$R_{\text{NS,Punc}}=\frac{\sum_j\lambda_j/j-\sum_r\rho_r/(r+2)}{\sum_j\lambda_j/j(1-P_0)+\sum_r\rho_r\frac{2^r-2}{r+2}}. \qquad \text{(C8)}$$

If $P_0\neq 1$, $R_{\text{NS,Punc}}$ can be written as

$$R_{\text{NS,Punc}}=\frac{1}{1-P_0}-\frac{\sum_r\frac{\rho_r(\frac{2^r-2}{1-P_0}+1)}{r+2}}{\sum_j\lambda_j/j(1-P_0)+\sum_r\rho_r\frac{2^r-2}{r+2}}. \qquad \text{(C9)}$$

For a check node of order r, the puncture rate is set as $r/(r+2)$. The total number of punctured edges is given by $\sum_r\frac{n\rho_r r}{r+2}$. Then we have

$$\sum_j n\lambda_j P_0=\sum_r\frac{n\rho_r r}{r+2}$$

$$\Rightarrow P_0=\sum_r\frac{\rho_r r}{r+2}. \qquad \text{(C10)}$$

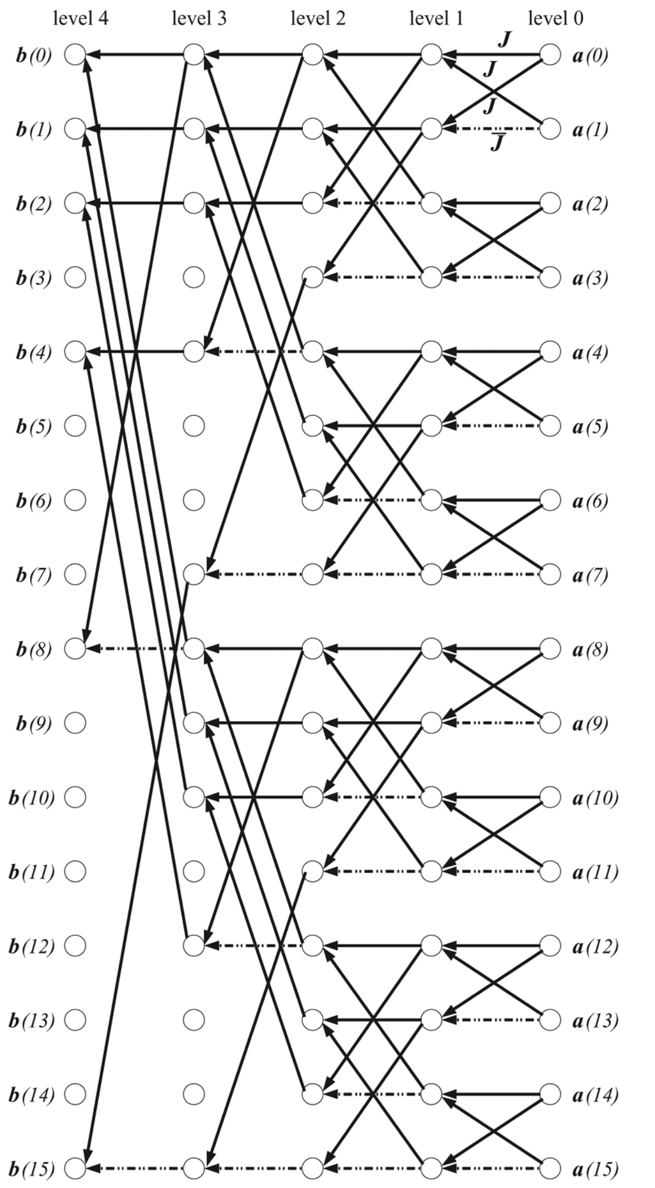


Fig. 15. Flow graph of length-16 reduced output APP-FHT, where only the branches leading to the $r + 2$ outputs are kept.

APPENDIX D
REDUCED APP FAST HADAMARD TRANSFORMS

In the LDPC-Hadamard decoding, we only need to compute $r + 2$ extrinsic messages at each check node. Therefore, the outputs of APP-FHT are required only at the information positions and the last bit. Define the index set $J_r \triangleq \{0, 1, 2, 4, \ldots, 2^{r-1}\}$. As shown in Fig. 15, the additions are only required for the bits/edges with indexes $\{k2^j + i | i \in J_j, k = 0, 1, 2, \ldots, 2^{r-j} - 1\}$ at level j. Note that in each transition, the nodes in Fig. 15 need two additions. Therefore, the total number of these dual-additions involved is

$$\begin{aligned} &2 \times 2^{r-1} + 4 \times 2^{r-2} + 5 \times 2^{r-3} + \cdots + (r+2) \times 2^0 \\ &= 3(2^r - 1) + 2^{r-1} - r - 1 \approx 3 \times 2^r. \end{aligned} \tag{D1}$$

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting and decoding: Turbo codes," in *Proc. Int. Conf. Commun. (ICC)*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[2] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 33, no. 6, pp. 457–458, Mar. 1997.

[3] R. G. Gallager, "Low-density parity check codes," *IRE Trans. Inf. Theory*, vol. 39, no. 1, pp. 37–45, Jan. 1962.

[4] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Analysis of low-density codes and improved designs using irregular graphs," in *Proc. ACM Symp. Theory Comp.*, Dallas, TX, 1998, pp. 249–258.

[5] S. Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity check codes using a gaussian approximation," *IEEE Trans. Inf. Theory*, vol. 47, pp. 657–670, Feb. 2001.

[6] T. Richardson, M. A. Shokrohalli, and R. Urbanke, "Design of capacity approaching irregular low density parity check codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 619–637, Feb. 2001.

[7] T. Richardson and R. Urbanke, "Capacity of low density parity check codes under message passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, pp. 599–618, Feb. 2001.

[8] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. 27, pp. 533–547, Sep. 1981.

[9] S. Dolinar, "Design and iterative decoding of networks of many small codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Yokohama, Japan, Jun.–Jul. 2003, pp. 346–346.

[10] L. Lentmaier and K. S. Zigangirov, "Iterative decoding of generalized low-density parity-check codes," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, Cambridge, MA, Aug. 1999, pp. 149–149.

[11] ——, "On generalized low-density parity-check codes based on hamming component codes," *IEEE Commun. Lett.*, vol. 3, pp. 248–250, Aug. 1999.

[12] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Trans. Inf. Theory*, vol. 50, pp. 2657–2673, Nov. 2004.

[13] I. B. Djordjevic, O. Milenkovic, and B. Vasic, "Generalized low-density parity-check codes for optical communication systems," *J. Lightw. Technol.*, vol. 23, no. 5, pp. 1939–1946, May 2005.

[14] S. ten Brink, "Convergence behavior of iterative decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, pp. 1727–1737, Oct. 2001.

[15] I. B. Djordjevic and B. Vasic, "Iteratively decodable codes from orthogonal arrays for optical communication systems," *IEEE Commun. Lett.*, vol. 9, pp. 924–926, Oct. 2005.

[16] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Trans. Inf. Theory*, vol. 47, pp. 2711–2736, Nov. 2001.

[17] B. Vasic, E. M. Kurtas, and A. V. Kuznetsov, "LDPC codes based on mutually orthogonal latin rectangles and their application in perpendicular magnetic recording," *IEEE Trans. Magn.*, vol. 38, no. 5, pp. 2346–2348, Sep. 2002.

[18] L. Ping, W. K. Leung, and K. Y. Wu, "Low-rate turbo-Hadamard codes," *IEEE Trans. Inf. Theory*, vol. 49, pp. 3213–3224, Dec. 2003.

[19] O. Wintzell, M. Lentmaier, and K. Zigangirov, "Asymptotic analysis of superorthogonal turbo codes," *IEEE Trans. Inf. Theory*, vol. 49, pp. 253–258, Jan. 2003.

[20] J. Hamkins and D. Divsalar, "Coupled receiver-decoders for low rate turbo codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Yokohama, Japan, Jun. 2003, pp. 381–381.

[21] A. J. Viterbi, "Very low rate convolutional codes for maximum theoretical performance of spread-spectrum multiple-access channels," *IEEE J. Sel. Area Commun.*, vol. 8, no. 4, pp. 641–649, Aug. 1990.

[22] L. Ping, L. H. Liu, K. Y. Wu, and W. K. Leung, "Approaching the capacity of multiple access channels using interleaved low-rate codes," *IEEE Commun. Lett.*, vol. 8, pp. 4–6, Jan. 2004.

[23] L. Yang and G. B. Giannakis, "Ultra-wideband communications: An idea whose time has come," *IEEE Sig. Proc. Mag.*, vol. 21, no. 6, pp. 26–54, Nov. 2004.

[24] J. Boutros, O. Pothier, and G. Zemor, "Generalized low density (Tanner) codes," in *Proc. of Int. Conf. Commun. (ICC)*, Vancouver, BC, Canada, Jun. 1999, pp. 441–445.

[25] Y. Wang and M. Fossorier, "Doubly generalized LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Seattle, WA, Jul. 2006.

[26] D. Raghavarao, *Constructions and Combinatorial Problems in Design of Experiments*. New York: Dover, 1971.

[27] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: A model and two properties," in *Proc. Conf. Inf. Sci. Syst. (CISS)*, Princeton, NJ, Mar. 2002.

[28] S. ten Brink and G. Kramer, "Design of repeat-accumulate codes for iterative detection and decoding," *IEEE Trans. Signal Process.*, vol. 51, pp. 2764–2772, Nov. 2003.

[29] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, pp. 670–678, Apr. 2004.

[30] Optimization of LDPC Codes [Online]. Available: http://lthcwww.epfl.ch/research/ldpcopt/

[31] A. Ashikhmin and S. Litsyn, "Simple MAP decoding of first-order reed-muller and hamming codes," *IEEE Trans. Inf. Theory*, vol. 50, pp. 1812–1818, Aug. 2004.

[32] J. K. Wolf, "Efficient maximum likelihood decoding of linear block codes using a trellis," *IEEE Trans. Inf. Theory*, vol. 24, pp. 76–80, Jan. 1978.

[33] M. G. Luby, M. Mitzenmacher, M. A. Shokrohalli, and D. A. Spielman, "Improved low density parity-check codes using irregular graphs," *IEEE Trans. Inf. Theory*, vol. 47, pp. 585–598, Feb. 2001.

[34] M. Tüchler and J. Hagenauer, "Exit charts and irregular codes," in *Proc. Conf. Info. Sci. Syst. (CISS)*, Princeton, NJ, Mar. 2002.

[35] G. Yue and X. Wang, "Optimization of irregular repeat accumulate codes for MIMO systems with iterative receivers," *IEEE Trans. Wireless Commun.*, vol. 4, no. 6, pp. 2843–2855, Nov. 2005.

[36] P. Komulainen and K. Pehkonen, "Performance evaluation of superorthogonal turbo codes in AWGN and flat Rayleigh fading channels," *IEEE J. Sel. Areas Commun.*, vol. 16, pp. 196–205, Feb. 1998.