Results on Punctured Low-Density Parity-Check Codes and Improved Iterative Decoding Techniques

Hossein Pishro-Nik, *Member, IEEE*, and Faramarz Fekri, *Senior Member, IEEE*

***Abstract*—This paper first introduces an improved decoding algorithm for low-density parity-check (LDPC) codes over binary-input–output-symmetric memoryless channels. Then some fundamental properties of punctured LDPC codes are presented. It is proved that for any ensemble of LDPC codes, there exists a puncturing threshold. It is then proved that for any rates R_1 and R_2 satisfying $0 < R_1 < R_2 < 1$, there exists an ensemble of LDPC codes with the following property. The ensemble can be punctured from rate R_1 to R_2 resulting in asymptotically good codes for all rates $R_1 \leq R \leq R_2$. Specifically, this implies that rates arbitrarily close to one are achievable via puncturing. Bounds on the performance of punctured LDPC codes are also presented. It is also shown that punctured LDPC codes are as good as ordinary LDPC codes. For BEC and arbitrary positive numbers $R_1 < R_2 < 1$, the existence of the sequences of punctured LDPC codes that are capacity-achieving for all rates $R_1 \leq R \leq R_2$ is shown. Based on the above observations, a method is proposed to design good punctured LDPC codes over a broad range of rates. Finally, it is shown that the results of this paper may be used for the proof of the existence of the capacity-achieving LDPC codes over binary-input–output-symmetric memoryless channels.**

***Index Terms*—Bipartite graphs, capacity-achieving codes, erasure channel, improved decoding, low-density parity-check (LDPC) codes, iterative decoding, punctured codes, rate-adaptive codes, rate-compatible codes, symmetric channels.**

I. Introduction

We study some coding schemes using low-density parity-check (LDPC) codes. We specifically concentrate on improved decoding methods and rate-compatible LDPC codes. One of the most important problems in LDPC coding is the analysis and design of finite-length codes. Although there has been some results on finite-length analysis of LDPC codes over the binary erasure channel (BEC) [1], design of good finite-length LDPC codes is still a research problem. An alternative way to compensate for this problem is to improve the decoding of these codes. Some previous work on improved decoding algorithms can be found in [2] and [3]. In [4] we introduced an improved decoding algorithm for LDPC codes over the BEC. We showed that the proposed decoding method can decrease the error rate of the code by several orders of magnitude on BEC with very little extra complexity. In this paper, we generalize the algorithm for arbitrary binary-input–output-symmetric memoryless (BIOSM) channels. For the binary input additive white Gaussian noise channel and an LDPC code of length 1000, we show that the algorithm results in.5 dB gain with respect to the belief propagation. We also show that the proposed method can be made more effective on nonuniform channels.

We then study punctured LDPC codes and show some fundamental properties of these codes. We prove that punctured LDPC codes have a threshold effect and compute the threshold for different methods of puncturing. We specifically show that arbitrary rates are achievable via puncturing. We then discuss the optimality of punctured LDPC codes. Much stronger results can be showed for the BEC. For example, using only one encoder and decoder we can achieve the capacity of BEC on arbitrary set of rates. We discuss design of good puncturing schemes for LDPC codes and we propose a simple rule for constructing rate-compatible LDPC codes. The proposed method prevents the performance degradation for the high rates that was previously observed in [5]. It is also applicable to finite-length LDPC codes. Finally we consider capacity achieving sequences for general BIOSM channels. We prove that if capacity achieving sequences of LDPC codes exist when the rate of the codes approaches zero, then capacity achieving LDPC codes exist for all rates.

Throughout the paper we adopt the following terminology. By a graph we mean a simple graph, i.e., a graph with no loops (edges joining a vertex to itself) and no multiple edges (several edges joining the same two vertices). Let A be a subset of the vertices in the graph g. $N(A) = N^1(A)$ shows the set of neighbors of A in g. More generally, for $j \in \mathbb{N}$, $N^j(A)$ is the set of vertices in g from which there is path of length j to a vertex in A. Let D be a subgraph of g such that its vertex set is A. We say D is induced by A if D contains all edges of g that join two vertices in A. For a graph g, $deg_g(v)$ is the degree of v in g. If V is the set of vertices in g and $U \subseteq V$, then $deg_U(v)$ is the number of neighbors of v in U. For a random variable X, we show its distribution by $F_X(x)$. If the random variable has a well-defined density function, we represent the density function by $f_X(x)$. Similar to [6], we define $P_e(F_X) = \Pr\{X < 0\} + \frac{1}{2}\Pr\{X = 0\}$.

II. Improved Decoding Algorithms for Finite-Length LDPC Codes

The iterative decoding of LDPC codes is very fast especially for the BEC. Our aim in this section is to improve the error performance of iterative decoding algorithm while keeping the

Manuscript received August 6, 2003; revised February 23, 2005. This work was supported by the National Science Foundation under Grants ANI-0117840 and CCF-0430964. The material in this paper was presented in part at the IEEE Information Theory Workshop, San Antonio, TX, October 2004.

H. Pishro-Nik is with the School of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003-9292 USA (e-mail: pishro@ecs.umass.edu).

F. Fekri is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250 USA (e-mail: fekri@ece.gatech.edu).

Communicated by R. Urbanke, Associate Editor for Coding Techniques.

Digital Object Identifier 10.1109/TIT.2006.889701

decoding fast. We use the message passing algorithm with some modifications. Let $T_A(n)$ be the average time required for the standard iterative decoding (Algorithm A) of an LDPC code of length n when it is used over a BIOSM channel. Let B be an improved decoding method for the same code when used over the same channel. Let $T_B(y,n)$ be the time that Algorithm B needs to decode a received word y and let $T_B(n)$ be the average time of the decoding of the code using Algorithm B. We want to have

$$T_B(n) \leq (1+\gamma)T_A(n) \quad (1)$$

$$\forall\, y, \quad T_B(y,n) \leq CT_A(n) \quad (2)$$

where γ is a small constant close to zero and C is a sufficiently small constant. For the BEC, our simulations show that the algorithm we propose (Algorithm C) achieves the above inequalities for $\gamma < 0.05$ and $C < 7$ when the length of the code is several thousands ($n \leq 5000$). For the general BIOSM channels, the proposed algorithm (Algorithm D) satisfies the above inequalities for $\gamma < .05$ and $C < 40$. Thus, for the BEC both average and maximum running time are small enough. For other BIOSM channels the average running time is still almost the same as the iterative decoding but the maximum running time can be 40 times the average running time of the standard decoding. However, for both channels, the proposed algorithms result in considerable reduction of the bit error rate with respect to the standard iterative decoding. For convenience, we first briefly review the proposed method for BEC.

A. Binary Erasure Channel (BEC)

In this section we briefly review the improved decoding algorithm for LDPC codes over the BEC [4]. The key idea can be summarized by the following observation. Consider a BEC with an erasure probability ϵ and an LDPC code of length n that has small enough error probability. If the message passing decoder fails to decode a received word completely, then there exists a very small number (usually less than or equal to 3 bits) of undecoded bits that by knowing their values, the decoder can finish the decoding successfully.

The message passing decoding of LDPC codes over the BEC can be described as follows [7].

- For all unlabeled check nodes do the following. If the values of all but one of the variable nodes connected to the check node are known, set the missing variable bit to the XOR of the other variable nodes and label that check node "finished." If all the variable nodes that are connected to the check node are known label the check node as finished.
- Continue the above procedure until all check nodes are labeled as finished or the decoding cannot continue further.

We call the above decoding method Algorithm A. We also call the first improvement to A as Algorithm B. For the erasure patterns that Algorithm A finishes the decoding successfully, both algorithms are the same. The difference between the two algorithms is when Algorithm A fails to complete the decoding of a received codeword. In this case Algorithm B chooses one of the undecoded variable nodes w_1 and guesses its value (for example by setting its value to zero). Then it continues as following.

- For all unlabeled check nodes do the following. If the values of all but one of the variable nodes connected to the check node are known, set the missing variable bit equal to the XOR of the other variable nodes and label it as a finished check node. If all the variable nodes connected to the check node are known then if the check node is satisfied label that check node "finished." Otherwise label it as "contradicted."
- Continue the above procedure until all check nodes are labeled or the decoding cannot continue further.

Once the above procedure is finished, if all of the check nodes are labeled and none of them is labeled contradicted, the decoder outputs the resulting word as the decoded word. If all of the check nodes are labeled but some of them are labeled contradicted, then it changes the value of w_1, the guessed variable node, and repeats the decoding from there. This time the decoding finishes successfully because we have found the actual value of w_1. But if the decoding stops again (i.e., some of the check nodes are not labeled) we have to choose another unknown variable node w_2 and guess its value. Again, if some check nodes are labeled as contradicted, we have to go back and try other values for w_1 and w_2. Algorithm B is efficient only if the number of guesses is very small. Fortunately, simulation results show that even if we limit the number of guesses to a very small number, we can decrease the error rate considerably. Thus, in practice we limit the number of guesses to a maximum value $g_{\max}$ and we claim a decoding failure if the decoder requires more guesses. In fact, simulations show that with the right choices of the variable nodes to guess, usually the decoding finishes successfully by one or two guesses. Note that the above algorithm does not need any computations other than repeating the iterative decoding. Thus the decoding is very fast.

Algorithm B has two problems. First, the complexity of the algorithm grows exponentially with the number of guesses. Although the number of guesses is very small, this is undesirable. In fact, if the complexity of the algorithm increased linearly with the number of guesses we could increase $g_{\max}$ and decrease the error probability substantially. Second, it is possible that the algorithm declares a wrong word as the output of the decoding. However, this can happen only if the maximum-likelihood (ML) decoder cannot decode the corresponding codeword. Since the ML decoder has a very low error probability, this happens with a very small probability.

As it is shown in [4] this algorithm can be significantly improved to get a new algorithm that we called Algorithm C. In Algorithm C, instead of trying different values for the guessed variable nodes, we can find these values efficiently. We showed that Algorithm C has very low complexity and it avoids the undetected errors. It is also shown in [4] that both Algorithms B and C result in considerable improvement upon the standard iterative decoding. It is also shown in [4] that using two-edge-connected components in the Tanner graph we can decrease the number of guesses in Algorithms B and C.

In Section II-B, we can generalize Algorithm B for other BIOSM channels. It seems that generalizing Algorithm C for other channels is impossible, because Algorithm C takes advantage of the special structure of the BEC. In fact, this explains the efficiency of Algorithm C.

B. BIOSM Channels

In this section, we generalize the improved decoding algorithms [4] for other BIOSM channels. Suppose that an LDPC code is used for error correction over a BIOSM channel. Let $V = V(g) = \{v_1, v_2, \ldots, v_n\}$ and $C = C(g) = \{c_1, c_2, \ldots, c_m\}$ be the sets of variable and check nodes in the Tanner graph of the code, respectively. Moreover, suppose we use the standard iterative decoding (Algorithm A) to decode the received words. Assume that Algorithm A has small enough error probability (for example less than 10^{-2}). The iterative decoder is initialized by the log likelihood ratio (LLR) of the variable nodes based on the observation of the channel output.

As we discussed Algorithms B and C in the previous section were based on the following observation. When the iterative decoding fails, knowing the values of a few bits in the stopping set is sufficient to finish the decoding successfully. We first extend this observation to arbitrary BIOSM channels.

Suppose a codeword $\underline{X} = (x_1, x_2, \ldots, x_n)$ is transmitted over the channel, where n is the code length. We define a function $\ell : \{-1, +1\} \longmapsto \{-\infty, +\infty\}$ as

$$\ell(x) = \begin{cases} +\infty & \text{if } x = +1 \\ -\infty & \text{if } x = -1. \end{cases} \tag{3}$$

Let $\underline{L} = (l_1, l_2, \ldots, l_n)$ be the (LLR) of the corresponding variable nodes $v_1, v_2, \ldots, v_n$ based on the observation of the channel output. Suppose the iterative decoder fails to decode a received word. In other words, after the maximum number of iterations, there are still unsatisfied check nodes in the graph. In this case there exists a set $I \subset [n] = \{1, 2, \ldots, n\}$ with a very small cardinality $|I|$ (usually $|I| \leq 4$) that has the following property.

Define $\underline{L'} = (l'_1, l'_2, \ldots, l'_n)$ as

$$l'_i = \begin{cases} +l_i & \text{if } i \in [n] - I \\ \ell(x_i) & \text{if } i \in I. \end{cases} \tag{4}$$

Now, if we initialize the iterative decoder with $\underline{L'}$, it can decode the received word correctly. We do not have proof for this statement. However, the decoding algorithm that we will introduce works well in practice.

Based on the above development, we design a decoding algorithm similar to Algorithm B that was presented for the BEC [4]. We call it Algorithm D. Recall that in Algorithm B, we chose the bits to be guessed from the stopping set after the decoding failure. For the general case of BIOSM channels, we need to find a method to choose the variable nodes whose values must be guessed.

We observed that the following simple method works very well in practice for choosing the variable nodes to be guessed. Let m_i be the number of unsatisfied check nodes that are adjacent to v_i when the iterative decoding fails. We choose the variable nodes that have the highest m_i's. Intuitively, by this method we find the locations of the graph for which there is a lack of information. Choosing variable nodes with high m_i, reduces the number of guesses required for successful decoding. In fact, if we just select the guessed nodes randomly, some of the guesses will not be necessary for successful decoding. Since the complexity of the algorithm increases exponentially with respect to the number of guesses, it is important to have as few guesses as possible. Note that a variable node that we choose to guess may have the correct value at the end of the iterative decoding. However, it is connected to several unsatisfied check nodes. Since we set the LLR of the guessed variable node to $+\infty$ or $-\infty$, this can help the iterative decoder to correct the values of the other bits connected to the check nodes. Again, we need to choose a maximum value for the number of guesses. Empirical observations show that choosing the maximum number of guesses as five can reduce the bit error rate considerably. Note that, similar to the case of the binary erasure channel (Algorithms B and C), the average running time of Algorithm D, $T_D(n)$, is almost the same as Algorithm A (the standard iterative decoding). However, to maintain the maximum running time small enough we need to choose a small value for the maximum number of guesses.

We summarize Algorithm D as follows. For any received codeword we perform the standard iterative decoding. If all check nodes are satisfied at the end of decoding, we are done. Otherwise, we find g variable nodes that are connected to highest number of unsatisfied check nodes at the end of the decoding and guess their values. We repeat the standard decoding but this time we initiate the algorithm with the new LLRs for the guessed bits ($+\infty$ or $-\infty$). For other bits, we use the LLRs found by the channel observation. We repeat the above procedure until either the decoding finishes successfully or all 2^g values for the guessed variable nodes are tried.

To evaluate the performance of Algorithm D we chose an LDPC code of length 1000 with the following degree distribution:

$$\begin{aligned} \lambda_1(x) &= 0.1212x + 0.6364x^2 + 0.2424x^7 \\ \rho_1(x) &= 0.3818x^5 + 0.5939x^6 + 0.0243x^7. \end{aligned} \tag{5}$$

We used the expurgated ensemble. That is, we generate a code from the ensemble, and if the minimum distance of the code is small, we do not use the code and pick another code at random. Since the ensemble has asymptotically linear minimum distance [8], [9], after a few tries we will find a code with large minimum distance. We obtained the bit error rate performance for both Algorithms A (the standard iterative decoding) and D. For Algorithm D, we chose the number of guesses $g = 5$. Fig. 1 shows the performance of the decoders. We observe that Algorithm D has 35 dB gain with respect to Algorithm A at the bit error rate of 10^{-5}. The gain increases to.5 dB at the bit error rate of 10^{-6}. The figure also shows the performance of a randomly chosen code of length 1000 from the ensemble of $(3, 6)$ regular codes, which is known to have the best performance among the regular LDPC code ensembles. We observe that using Algorithm D for decoding the above irregular code results in 1 dB gain over the $(3, 6)$ regular code in low bit error rates.

Now we present some experimental results concerning the running time of Algorithm D. We give the results for the code of length 1000 from the ensemble $g(\lambda_1, \rho_1)$. We decoded 10^9 bits over the binary-input additive white Gaussian noise (BI-AWGN) channel and measured the average and the maximum running time for the decoding of received blocks. Let $T_A(n)$ and $T_D(n)$ show the average time of decoding of the LDPC code of length n from the given ensemble using Algorithms

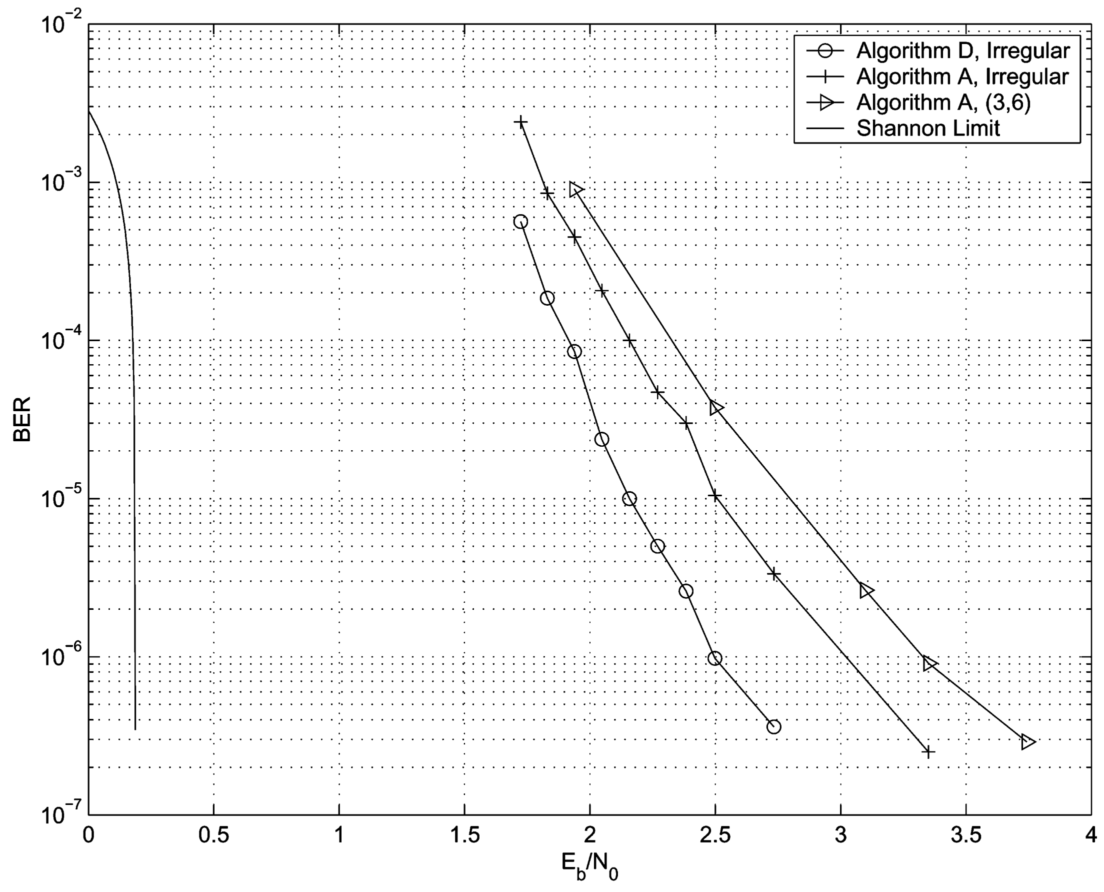

Fig. 1. Comparisons of the bit error rates of Algorithms A and D for an irregular code of length $n = 10^3$ and the $(3, 6)$ regular code decoded by Algorithm A over the BIAWGN channel.

A and D, respectively. Our simulations suggest that the estimate of $\frac{T_D(n)}{T_A(n)}$ is equal to 1.12. This implies that the average running time of Algorithms A and D are almost the same. Recall that we defined $T_D(y, n)$ as the time that Algorithm D needs to decode a received word y. Let $T_D^m(n)$ denote the maximum value of $T_D(y, n)$ over all the received blocks. We define $R_D(n) = \frac{T_D^m(n)}{T_A(n)}$. Our simulations show that $R_D(n) = 38.4$, approximately. In other words, in the worst case the time that Algorithm D needs to decode a received word can be 38.4 times the average running time of Algorithm A. In [4] we have estimated that $R_C(n) = 6.7$, for the BEC and a half-rate code with length 1000.

III. Non-Uniform Channels

In [10] and [11] we investigated the application of LDPC codes for nonuniform channels. We introduced a scheme for designing LDPC codes over these channels. We also showed that the punctured codes can be viewed as a special case of nonuniform channels. In this section we study some properties of LDPC codes on nonuniform channels and show that the improved decoding algorithm proposed in this paper is more effective on these channels if used properly. Specifically, we study the effect of the algorithm on punctured LDPC codes.

A nonuniform channel can be considered as several parallel independent subchannels as it is shown in Fig. 2. We assume that we use one LDPC code over the set of subchannels. Thus, different bits in a codeword may be transmitted over different subchannels. Some practical examples of nonuniform channels are volume holographic memory (VHM) systems, orthogonal division frequency multiplexing (OFDM) systems, and multilevel coding.

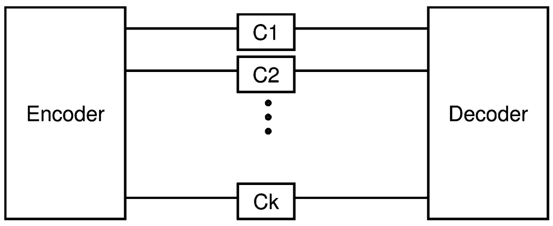

Fig. 2. Several parallel channels.

In VHM systems, the information is recorded and retrieved in the form of two-dimensional data pages, i.e., two-dimensional patterns of bits. These bits are subject to different sources of noise. The SNR decreases as we move from the center to the corner of the page. Typically, raw bit error rate might vary by two or three orders of magnitude over a page. As we explained in [11] we can divide a VHM page to k regions R_i such that bits of the same region have almost the same raw error probability. Any region in the page corresponds to one of the subchannels in Fig. 2. A similar situation exists in OFDM systems that consist of several parallel channels with different SNRs.

Suppose we use a code of length n. We transmit every codeword such that $n^{(j)}$ bits from each codeword are transmitted over the jth channel. In [11] we defined an ensemble $g(\Lambda,\rho)$ of LDPC codes. We showed that they have some good properties. For convenience, we repeat the definition of the ensemble here. The main point is that in the ensemble $g(\Lambda,\rho)$, bits of different types may have different degree distributions. Formally, let $(x_1,x_2,\ldots,x_n)$ be a codeword. Let also $W^{(j)}$ be the set of bits from the codeword that are transmitted over the jth channel (type j bits). Thus we have $|W^{(j)}|=n^{(j)}$, where $|\cdot|$ denotes the cardinality of the set. For example, in the VHM system, $W^{(j)}$ is the set of bits in the jth region (i.e., $W^{(j)}=\{x_i : x_i\in R_j\}$). Now we define the ensemble $g(\Lambda,\rho)$ of bipartite graphs that we propose for nonuniform error protection. Let E be the set of edges in the graph and let $E^{(j)}$ be the set of edges that are incident with a variable node of type j. Also let $E_i^{(j)}$ be the set of edges adjacent to the variable nodes of type j and degree i. We define

$$\lambda^{(j)}(x)=\sum\lambda_i^{(j)}x^{i-1} \tag{6}$$

where

$$\lambda_i^{(j)}=\frac{\left|E_i^{(j)}\right|}{|E^{(j)}|}. \tag{7}$$

Let $\Lambda=\{\lambda^{(j)}(x) : j=1,\ldots,k_r\}$. Let also $\rho(x)=\sum\rho_i x^{i-1}$ be as defined in [6]. We define the ensemble $g(\Lambda,\rho)$ as the ensemble of bipartite graphs with the degree distributions given by Λ and ρ.

Now we give some properties of the densities of the messages in the belief propagation algorithms on the ensemble $g(\Lambda,\rho)$. These properties are specifically useful for applying the improved decoding algorithm. Let $m_{vc}^{(l),(j)}$ denote the message that is sent from a variable node of type j (i.e., $v\in W^{(j)}$) to its incident check node c in the l'th iteration of the message passing algorithm. Let also $m_{cv}^{(l)}$ denote the message that the check node c sends to its incident variable node. Let $F_l^{(j)}$ and Q_l denote the average asymptotic distributions of random variables $m_{vc}^{(j),(l)}$ and $m_{cv}^{(l)}$, respectively.

Consider a BIOSM channel with parameter θ, where $\theta\in[\theta_{\min},\theta_{\max}]$ and $\theta_{\min},\theta_{\max}\in\mathbb{R}\cup\{-\infty,+\infty\}$. For example, for the BIAWGN channel, θ can be considered as the variance σ of the noise. Let $\mathcal{C}$ be a class of channels with parameter θ. Thus, any channel C_θ in $\mathcal{C}$ in is uniquely determined by its variable θ. A channel in $\mathcal{C}$ with parameter θ_0 is called C_{θ_0}. The capacity of the channel C_{θ_0} is shown by c_{θ_0}. Similar to [12], we consider physically degraded channels. For clarity of exposition we assume that if $\theta_1<\theta_2$, then C_{θ_2} is physically degraded with respect to C_{θ_1}.

Consider the case that in Fig. 2 all subchannels are the same type but have different channel parameter. Moreover, all subchannels belong to a class of physically degraded channels $\mathcal{C}$ as explained above. Suppose we use an LDPC code from the ensemble $g(\Lambda,\rho)$ over these channels. Assume the variable nodes of type j are transmitted through the channel C_{θ_j}. Then we have the following theorem:

Theorem 1: If $\theta_i<\theta_j$ and $\lambda^{(i)}(x)\equiv\lambda^{(j)}(x)$, then for any l we have $P_e\left(F_l^{(i)}\right)\le P_e\left(F_l^{(j)}\right)$.

Proof: This theorem can be proved using similar discussions as in [12]. Let u_i be a variable node of type i in the Tanner graph of the code. Thus, u_i receives observation from the output of the channel C_{θ_i}. Let

$$B_l=\bigcup_{k=0}^{2l}N^k(u_i) \tag{8}$$

be the neighborhood of u_i of depth l. Let g_B be the graph induced by the vertices in B_l. In the graph g_B, the variable node u_i receives information from the channel C_{θ_i}, and other variable nodes receive observation from possibly different channels. Let O_i be the observation of u_i and let O be the set of observations of the other variable nodes in B_l. Now, assuming that g_B is a tree, $P_e\left(F_l^{(i)}\right)$ is the error probability of the ML decoder based on the observations of the variable nodes in B_l (i.e., $O\cup\{O_i\}$). Now, in the above graph, if we just replace the variable node u_i with the variable node u_j that receives observation from the channel C_{θ_j}, again $P_e\left(F_l^{(j)}\right)$ is the error probability of the ML decoder based on the observations of the variable nodes in B_l (i.e., $O\cup\{O_j\}$). Since C_{θ_j} is physically degraded with respect to C_{θ_i}, we can consider O_j as the result of passing O_i through another channel C'. Since given O_i, the observation O_j is independent of the value of the transmitted bit, $P_e\left(F_l^{(i)}\right)$ is the error probability of the ML decoder based on the observations $O\cup\{O_i\}\cup\{O_j\}$. Thus $P_e\left(F_l^{(i)}\right)\le P_e\left(F_l^{(j)}\right)$ for the given g_B. Since $\lambda^{(i)}(x)\equiv\lambda^{(j)}(x)$, any structure of the neighborhood (the graph g_B and the channels from which the bits receive information) has the same probability of occurrence for u_i and u_j. Thus we conclude that $P_e\left(F_l^{(i)}\right)\le P_e\left(F_l^{(j)}\right)$. $\square$

Theorem 1 states that, under certain conditions, the bits that have higher error probability before the decoding, have higher error probability after the decoding as well. We have the following corollary.

Corollary 1: In a regular ensemble, the variable nodes that receive information from channels with lower capacity, have higher error probability after the decoding.

In the following, we study applications of improved decoding (Algorithm D) on nonuniform channels. In the guessing process we choose a variable node to be guessed. As we mentioned, it is better to choose the variable node from the parts in the graph that there is a lack of information. Since the likelihood of the error for the nodes that receive information from the channels with smaller capacity is higher, one simple method in the guessing process is to give priority to these nodes. Fortunately, our simulations show that this simple method works very well and considerably improves the decoding performance.

Punctured codes can be considered as a special case of nonuniform channels in which the punctured bits are transmitted through a channel with zero capacity. In Section V we study this special case of nonuniform channels in greater detail. To observe the performance of Algorithm D on nonuniform channels, we chose a $(3,6)$ regular LDPC code of length 1000.

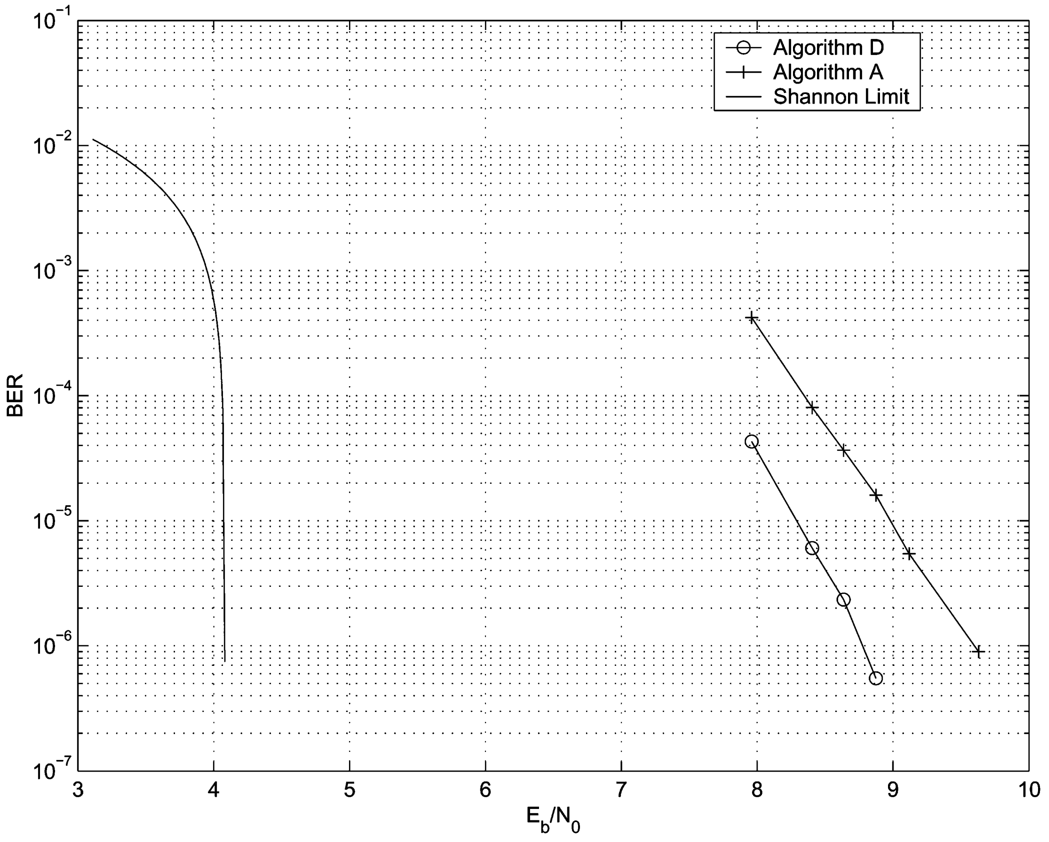

Fig. 3. Comparisons of the bit error rates of Algorithms A and D for the $(3, 6)$-regular LDPC code over the BIAWGN channel.

We randomly chose 37.5% of the variable nodes (i.e., 375 variable nodes) and designate them as punctured variable nodes. Thus the resulting code has the rate 0.8. We then evaluate the performance of Algorithms A and D for this code. As discussed above, in choosing the variable nodes to guess, the punctured bits were given priority. Fig. 3 shows the performance of the algorithms. We observe that improvements close to 1 dB is gained at low bit error rates using Algorithm D.

IV. Application of Pseudo-Codewords to the Analysis of Algorithm D

Simulation results given in previous sections suggest that we can considerably improve the performance of LDPC codes using Algorithm D. However, so far our results are totally based on simulations. In this section, we provide some discussions based on pseudocodewords [13], [14], [15] that help us understand Algorithm D. We first briefly provide some definitions and results from [15]. For more details, readers are referred to [15]. Let $G = (V, E)$ be a graph with vertex set $V = V(G) = \{v_1, v_2, \dots, v_l\}$ and edge set E. Finite covers are defined in [15] in the following way.

Definition 1: A finite degree m cover of $G = (V, E)$ is a graph $\hat{G}$ with vertex set $\hat{V} = \bigcup_{i=1}^{l} \hat{V}_i$, where each set $\hat{V}_i = \{\hat{v}_{i,1}, \hat{v}_{i,2}, \dots, \hat{v}_{i,m}\}$ contains m vertices. The edge set $\hat{E}$ of $\hat{G}$ is chosen as a subset of $\{\{\hat{v}_{i,s}, \hat{v}_{j,r}\} : \{v_i, v_j\} \in E, s, r \in \{1, 2, \dots, m\}\}$ such that for each vertex $\hat{v}_{i,s} \in \hat{V}$, we have $deg_{\hat{G}}(\hat{v}_{i,s}) = deg_G(v_i)$, and $|N(\hat{v}_{i,s})| = |N(v_i)|$. Moreover, $N(\hat{v}_{i,s})$ contains exactly one vertex $\hat{v}_{j,r}$ for all j for which $v_j \in N(v_i)$.

In simple terms, the graph $\hat{G}$ is obtained by replicating every vertex in G m times and introducing edges so that the local adjacency relationships between replicated nodes are preserved.

Let g be a Tanner graph of an LDPC code C. Let also $V = V(g) = \{v_1, v_2, \dots, v_n\}$ and $C = C(g) = \{c_1, c_2, \dots, c_m\}$ be the sets of variable and check nodes in the Tanner graph of the code, respectively. For simplicity, suppose we use the code C over a BIAWGN channel. Let $\hat{g}$ be a finite m cover of g and $\hat{C}$ be the corresponding code. For any codeword $\hat{c} \in \hat{C}$ a vector $w = w(\hat{c})$ is defined in [15], which is called a pseudocodeword of C. The fundamental cone of the graph g, denoted by $\mathcal{F}(g)$, is defined in [15] and is related to the set of pseudocodewords $w(\hat{c}) = w$ taken over all covers of g of all degrees $m = 1, 2, \dots$.

Let $\eta = (\eta_1, \eta_2, \dots, \eta_n)$, be the set of received LLRs from the channel. Assuming that all-one word was transmitted, it is shown in [15] that the decoding is successful if and only if for all $w \in \mathcal{F}(g)$, we have $\langle w, \eta \rangle > 0$.

Using the concept of pseudocodewords we can determine a necessary and sufficient condition for the success of Algorithm D. Assume $\eta = (\eta_1, \eta_2, \dots, \eta_n)$ be the LLR vector received from a BIAWGN channel. Moreover, assume that the standard iterative decoding has failed. Define

$$\mathfrak{P}(\eta) = \{w \in \mathcal{F}(g) : \langle w, \eta \rangle < 0\}. \tag{9}$$

The following simple Lemmas give necessary and sufficient conditions for the success of Algorithm D. Let

$\mathcal{G} = \{i : v_i \text{ is guessed}\}$ be the set of indices of the guessed variable nodes in Algorithm D. Let η' be the new LLRs imposed by Algorithm D (i.e., by replacing the LLR of the guessed bits by $-\infty$ or $+\infty$).

Lemma 1: Algorithm D succeeds only if for all $w \in \mathfrak{P}(\eta)$, there exists $i \in \mathcal{G}$, such that $w_i > 0$.

Proof: Suppose there exists $w^* \in \mathfrak{P}(\eta)$, such that for all $i \in \mathcal{G}$, we have $w_i^* = 0$. Then, $\langle w^*, \eta' \rangle = \langle w^*, \eta \rangle < 0$. Thus the iterative decoding fails independent of the values of the guessed bits. □

We now show that the condition of Lemma 2, is actually a sufficient condition for Algorithm D to converge to the ML decoding.

Lemma 2: If for all $w \in \mathfrak{P}(\eta)$, there exists $i \in \mathcal{G}$, such that $w_i > 0$, then Algorithm D can find the ML decoded codeword.

Proof: Assume the all-one codeword is transmitted. Since we check all possible values for the guessed bits, at some point we will guess the correct value for all the guessed bits. That is the LLRs for all the guessed bits become $+\infty$. Since for all $w \in \mathfrak{P}(\eta)$, there exists $i \in \mathcal{G}$, such that $w_i > 0$, we conclude that for all $w \in \mathfrak{P}(\eta)$, $\langle w \cdot \eta' \rangle = +\infty$. Thus the decoding is successful.

Therefore, when the guesses are correct the decoding is successful. On the other hand it is possible that the algorithm converges to a wrong codeword for some other guessed values. In fact, for good LDPC codes, this is very unlikely. However, in these situations at the end of decoding we will have more than one decoded valid codewords. Since the number of these codewords is small a simple ML test will give us the result of ML decoding. □

Using the above result one may try to analyze the performance of Algorithm D. In fact, this is one of the future directions of our research.

V. Punctured LDPC Codes

A. Introduction

In the previous section, we showed using the proposed decoding algorithm presented in this paper we can considerably improve the performance of punctured LDPC codes. In this section, we study some fundamental properties of punctured LDPC codes. Puncturing is one of the most common methods to construct rate-compatible codes. In this method, to change the rate of a code to a higher rate, we puncture (delete) a subset of the codeword bits. We first study the puncturing capacity of LDPC code ensembles. Particularly, we prove that any LDPC code ensemble has a puncturing threshold p^*. If the puncturing fraction p is smaller than p^*, then the punctured code is asymptotically good. In other words, a code from the ensemble can be used to achieve arbitrarily small error probability over a noisy channel while the code rate is bounded away from zero. On the other hand, if $p > p^*$, error probability is bounded away from zero, independent of the communication channel. The puncturing thresholds can be easily computed for both randomly and intentionally punctured LDPC codes. We also show that puncturing is a lossless process in the sense that for any ensemble of LDPC codes of rate R_1, there exists a punctured LDPC code ensemble of an arbitrary rate $R_2 < R_1$ with the same threshold under the message passing decoding algorithms. We then show that for any rates R_1 and R_2 satisfying $0 < R_1 < R_2 < 1$, there exists an LDPC code that can be punctured from rate R_1 to R_2 such that the resulting code is asymptotically good for all rates R, $R_1 \le R \le R_2$. Specifically, this shows that rates arbitrarily close to one are achievable using puncturing. For BEC, we show that using only one encoder and decoder and a proper puncturing scheme, we can achieve the capacity for an arbitrary set of positive rates. We then propose a method to design good punctured LDPC codes over a broad range of rates. The method is very simple and does not need any optimization process. Additionally, it avoids performance degradation at high rates. It is also applicable to finite-length LDPC codes. Finally, we show the possible application of punctured LDPC codes for proving the existence of capacity achieving sequences for BIOSM channels.

We consider the following scheme. We take an LDPC code of rate $R_p = k/n$, where k and n are the length of the information blocks and the codewords, respectively. To generate a code with a new rate, we puncture a subset of bits in the codeword and send the unpunctured bits to the receiver. It is assumed that the decoder knows the position of punctured bits in the codeword. To start the decoding, we need to compute LLRs in the decoder. The LLRs for the punctured bits are zero.

Ha and McLaughlin studied optimal puncturing of LDPC codes [5]. They studied two methods for puncturing LDPC codes. In the first method, the authors chose the punctured bits randomly (i.e., if the puncturing fraction is p, they chose a subset of the bits in the codeword with cardinality pn at random and puncture the bits in the subset). This method is called random puncturing. In the second method, the intentional puncturing, they optimized the puncturing distribution. The authors set up the intentional puncturing as follows. First, variable nodes of the bipartite graph were grouped in accordance with their degrees. Then, they randomly punctured a fraction π_j of the nodes in G_j, where G_j is the set of variable nodes of degree j.

As we mentioned, a punctured code can be modeled as a code that is used over two parallel channels as Fig. 2. In this model, punctured bits are transmitted over the second channel that has zero capacity. Let us define

$$\phi_j = \frac{\lambda'_j \pi_j}{\Sigma_j \lambda'_j \pi_j}, \quad p = \Sigma_j \lambda'_j \pi_j \tag{10}$$

where λ'_j is the fraction of degree-j variable nodes in the graph. Thus if we let $\Phi = \{\phi_j : j = 2, 3, \ldots, d_{v_{\max}}\}$, an intentional puncturing distribution for a code from (λ, ρ) ensemble can be represented by the pair (Φ, p) in which p shows the puncturing fraction and Φ determines the puncturing structure. A valid puncturing pattern is obtained when we have $p\phi_j \le \lambda'_j$, for $j = 2, 3, \ldots, d_{v_{\max}}$.

It is worth noting that asymptotically random puncturing is a special case of intentional puncturing, as stated in the following.

Fact 1: A randomly punctured ensemble of LDPC codes with a puncturing fraction p has the same threshold as the intention-

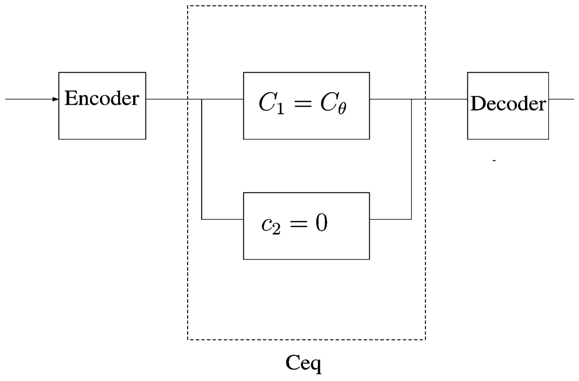

Fig. 4. Model that describes puncturing over a binary channel.

ally punctured code with puncturing distribution $\phi_j = \lambda'_j$, for $j = 2, 3, \ldots, d_{v_{\max}}$.

Proof: The proof is simple and can be done by evaluating the density evolution formulas for the punctured codes given in [11]. The proof alternatively follows from Strong Law of Large Numbers (as $n \longrightarrow \infty$, both of the distributions become the same almost surely), and using continuity of the density evolution over space of distributions. □

Fact 1 implies that any asymptotic result that is valid for intentionally punctured codes is usually valid for randomly punctured codes. Thus, whenever we are concerned with asymptotic properties, we only give the result for intentional puncturing. As we mentioned in [11] a punctured LDPC code can be modeled as Fig. 4. In this figure the punctured bits are transmitted through a channel with zero capacity. The actual channel, is a BIOSM channel with the parameter θ. When θ decreases, i.e., when the channel improves, we increase the puncturing fraction. Thus we can consider the puncturing process as a change in the channel not in the code rate.

B. Puncturing Threshold of LDPC Codes

In this section, we compute the puncturing threshold p^* of LDPC code ensembles. Consider an MBIOS channel with parameter θ, where $\theta \in [\theta_{\min}, \theta_{\max}]$ and $\theta_{\min}, \theta_{\max} \in \mathbb{R} \cup \{-\infty, +\infty\}$. For example, for the BIAWGN channel, θ can be considered as the variance σ of the noise. Let $\mathcal{C}$ be a class of channels with parameter θ. Thus, any channel C_θ in $\mathcal{C}$ in is uniquely determined by its variable θ. A channel in $\mathcal{C}$ with parameter θ_0 is called C_{θ_0}. The capacity of the channel C_{θ_0} is shown by c_{θ_0}. For simplicity, we assume that c_θ is a continuous function of θ. Similar to [12], we consider physically degraded channels. For clarity of exposition we assume that if $\theta_1 < \theta_2$, then C_{θ_2} is physically degraded with respect to C_{θ_1}. For the channel C_{θ_0} assuming the all-one code word has been sent, we define the random variable Z_{θ_0} as

$$Z_{\theta_0} = \ln \frac{p(X = 1|Y = y, \theta = \theta_0)}{p(X = -1|Y = y, \theta = \theta_0)} \tag{11}$$

where X and Y are the input and output of the channel, respectively. Assume that if $\theta_1 < \theta_2$, then C_{θ_2} is physically degraded with respect to C_{θ_1}. For simplicity, we assume that $P_e(F_{Z_\theta})$ is a continuous function of θ and we have $\lim_{\theta \to \theta_{\min}} P_e(F_{Z_\theta}) = 0$. Equivalently we can say that as θ tends to $\theta_{\min}$, Z_θ tends to infinity in probability [6] and thus the probability density function of Z_θ, f_{Z_θ}, converges to Δ_∞[6], [12]. Furthermore, we may assume that if $\theta > \theta_{\min}$, then $P_e(F_{Z_\theta}) > 0$. Note that almost all practical MBIOS channels such as BIAWGN channel, BSC, and BEC satisfy these properties.

We say an ensemble of LDPC codes of positive rate R is asymptotically good if there exists a $\theta_1 \in [\theta_{\min}, \theta_{\max}]$ such that $P_e(F_{Z_{\theta_1}}) > 0$ and a randomly chosen code from the ensemble can be used to achieve arbitrary small error rate over C_θ for all $\theta \leq \theta_1$. On the other hand, if we need $P_e(F_{Z_{\theta_1}}) \longrightarrow 0$ for achieving arbitrarily small error rate, the ensemble is said to be asymptotically bad. For a punctured ensemble (λ, ρ, Φ, p) we define

$$\lambda_i^{(2)} = \frac{\lambda'_i i \pi_i}{\Sigma \lambda'_j j \pi_j}, \quad q = \frac{\sum j \pi_j \lambda'_j}{\Sigma j \lambda'_j}. \tag{12}$$

We also define the following sequence:

$$x_0 = 1, \quad x_l = \lambda^{(2)}(1 - \rho(1 - q x_{l-1})). \tag{13}$$

Theorem 2: (Puncturing threshold of LDPC codes) Consider the ensemble of LDPC codes defined by the pair (λ, ρ) that are intentionally punctured by the puncturing pattern (Φ, p). Assume a randomly chosen code from the punctured ensemble is used over the channel C_θ, with $\theta > \theta_{\min}$. Let p_{th} be supremum value of the puncturing fraction p such that in (13), we have

$$\lim_{l \to \infty} x_l = 0. \tag{14}$$

If $p > p_{\text{th}}$, then the decoding error probability is bounded away from zero independent of the communication channel. On the other hand, if $p < p_{\text{th}}$, then there exists $\theta^* > \theta_{\min}$ such that if the channel parameter θ is smaller than θ^*, then

$$\lim_{l \to \infty} P_e(F_l) = 0. \tag{15}$$

The above result, proved in Appendix I, shows that p_{th} is the threshold of punctured LDPC codes. Using Theorem 2, we can find the puncturing threshold of randomly and intentionally punctured LDPC code ensembles. The cutoff rate of the code R_{th} is obtained by

$$R_{\text{th}} = \frac{R_p}{1 - p_{\text{th}}} \tag{16}$$

where R_P is the rate of the parent code. For example, since the $(3, 6)$ regular ensemble has puncturing threshold (for regular codes the random and intentional puncturing are the same) $p_{\text{th}} = .4294$, it has the cut off rate $R_{\text{th}} = 0.8763$. Thus using the $(3, 6)$ regular ensemble as the parent code ensemble, we cannot obtain the rates higher than 0.8763.

C. Achieving Arbitrary Rates Via Puncturing

In [5], authors evaluated the performance of several punctured LDPC codes and optimized the puncturing pattern to get the best performance. However, their simulations show that for high rates the performance of LDPC codes degrades and we

TABLE I
THE CUTOFF RATES OF SOME LDPC CODE ENSEMBLES (RANDOM PUNCTURING)

Ensemble	Cut Off Rates
(λ_2, ρ_2)	.944
(λ_3, ρ_3)	.9301
(λ_4, ρ_4)	.9463

need to pay a big penalty for using punctured codes. This phenomenon can be completely explained by the threshold effect of punctured codes discussed in the previous section.

In [5], authors used three ensembles of LDPC codes for puncturing. These ensembles are

$$\begin{aligned}\lambda_2(x) &= 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.4385x^9\\ \rho_2(x) &= 0.63676x^6 + 0.36324x^7\\ \lambda_3(x) &= 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6\\ &\quad + 0.30381x^{19}\\ \rho_3(x) &= 0.71875x^7 + 0.28125x^8\\ \lambda_4(x) &= 0.267817x + 0.204657x^2 + 0.077459x^5\\ &\quad + 0.2041810x^7 + 0.2458860x^{29}\\ \rho_4(x) &= x^5.\end{aligned}$$

The cut off rates of these ensembles are shown in Table I.

From Table I, we can easily explain the degradation in the performance of the above ensembles at high rates. By examining the simulation results in [5], we observe that all of the above ensembles show considerable degradation for the rates above 0.85. The degradation seems to have a very high slope for the rates above 0.9. This is because at these rates we are approaching the cut off rate.

Using the results of the previous section, by suitably choosing the ensemble, rates arbitrarily close to 1 can be obtained via puncturing. In fact, this is achieved by using good codes for the BEC as the parent code and randomly puncturing them (off course intentional puncturing works, as well). Since capacity achieving sequences of LDPC codes over the BEC are known [7], [16], [17] and [18], we can find codes with cut off rates arbitrarily close to 1. Moreover, the parent code can have any rate. Thus we have the following theorem.

Theorem 3: For any rates R_1 and R_2 that $0 < R_1 < R_2 < 1$, there exists an ensemble of LDPC codes with the following property. The ensemble can be punctured from rate R_1 to R_2 resulting in asymptotically good codes for all rates $R_1 \leq R \leq R_2$.

Theorem 3 assures that we can have a punctured LDPC codes that is asymptotically good (in the sense defined in this paper) on arbitrary set of rates; However, it does not give any clue how close to capacity the performance of such code will be. Here we give a lower bound on the achievable rates as the channel parameter changes.

Theorem 4: For any $\delta > 0$ and MBIOS channel C_θ parameterized by $\theta \in [\theta_{\min}, \theta_{\max}]$, and $R_1, R_2 \in (0,1)$, $R_1 < R_2$, there exists an ensemble (λ, ρ) of LDPC codes with the following properties. The ensemble can be punctured in a rate-compatible way to produce all rates $R \in [R_1, R_2]$. Moreover, for all $R \in [R_1, R_2]$ it can be used to reliably transmit data over C_{θ_R} satisfying

$$R > 1 - E[e^{(-Z_{\theta_R}/2)}] - \delta \tag{17}$$

where Z_{θ_R} is defined by (11).

Note that for any fixed rate, we can conclude the existence of an ordinary LDPC code ensemble that satisfies the above lower bound from [19]; however, the importance of Theorem 4 is in the fact that we can have only one code that simultaneously satisfies the bound for all rates.

Proof: Choose an LDPC code or rate R_1 whose threshold over the BEC, ϵ_{th}, satisfies

$$R_1 > 1 - \epsilon_{\text{th}} - \frac{R_1}{R_2}\delta. \tag{18}$$

We use this code as parent code and use random puncturing to obtain all rates R_p, $R_1 \leq R_p \leq R_2$. Now if $R_1 \leq R_p \leq R_2$, using Fig. 4, we can assume we still have a code of rate R_1, because as it was mentioned previously, puncturing can be considered as a change in the channel instead of the code rate. Now if we apply [19, Theorem 4.2] to this system, we conclude the following. If

$$p + (1-p)E[e^{(-Z_{\theta_{R_p}}/2)}] = \epsilon_{\text{th}} \tag{19}$$

then the threshold of the punctured code, θ_{th} satisfies $\theta_{\text{th}} \geq \theta_{R_p}$. Thus the punctured code can be used for reliable communication over $C_{\theta_{R_p}}$. However using [18], [19], and $R_1 \leq R_p = \frac{R_1}{1-p} \leq R_2$ we conclude

$$R_p > 1 - E[e^{(-Z_{\theta_{R_p}}/2)}] - \delta. \tag{20}$$

$\square$

Fig. 5 shows the ratio of the achievable rate and the channel capacity for BSC. Fig. 6 shows the gap from the capacity for BI-AWGN channel. As we see the gap is always less than 1.6 dB. The bound of Theorem 4 is interesting because it gives an analytical result; however, in practice we can find punctured LDPC codes that have better performance.

D. Optimality of Punctured LDPC Codes

In this section we show that by using punctured codes we do not lose performance. In other words, we show that for any LDPC ensemble of rate $R_1 > 0$ and any number R_2 satisfying $R_2 < R_1 < 1$, there exists an ensemble of punctured LDPC code of rate R_1 and parent rate R_2 with the same performance. We also propose a method to construct the punctured code with the same performance as a given code. Although these punctured codes have the same asymptotic performance as the unpunctured ones, they can have better finite-length performance.

Consider the parity check equation

$$c_1 : x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 = 0. \tag{21}$$

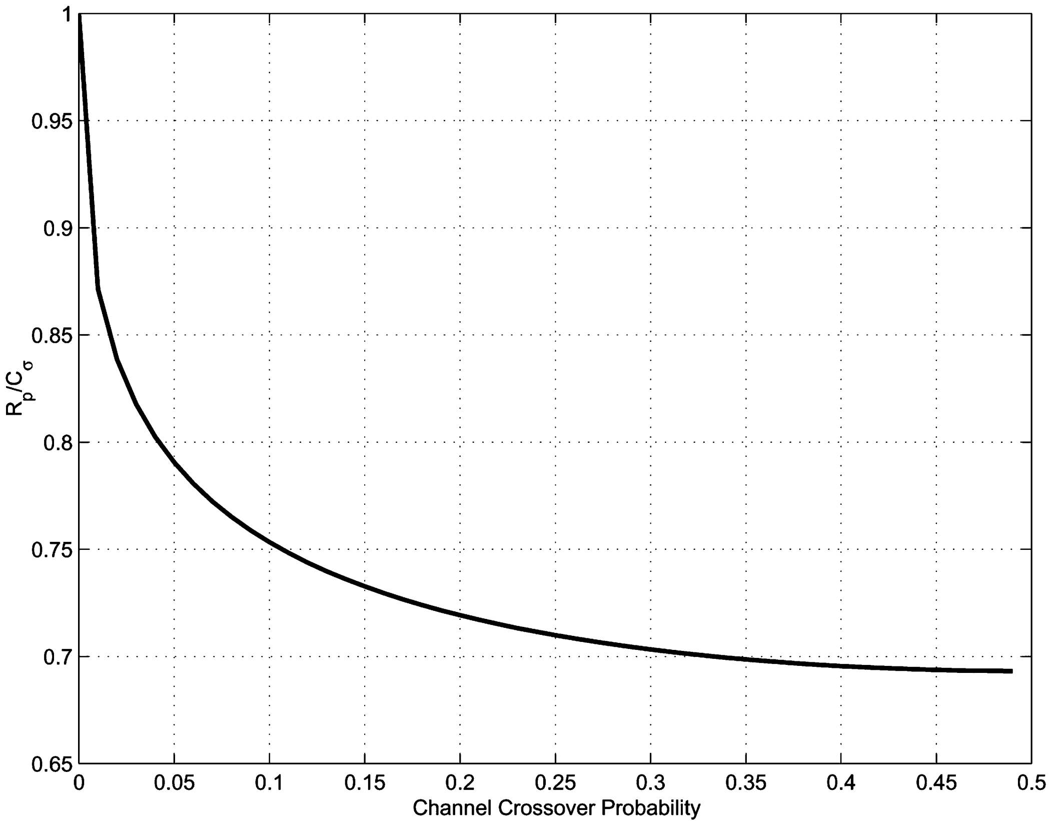

Fig. 5. The ratio of the achievable rate and the capacity for an ensemble of punctured LDPC codes over BSC.

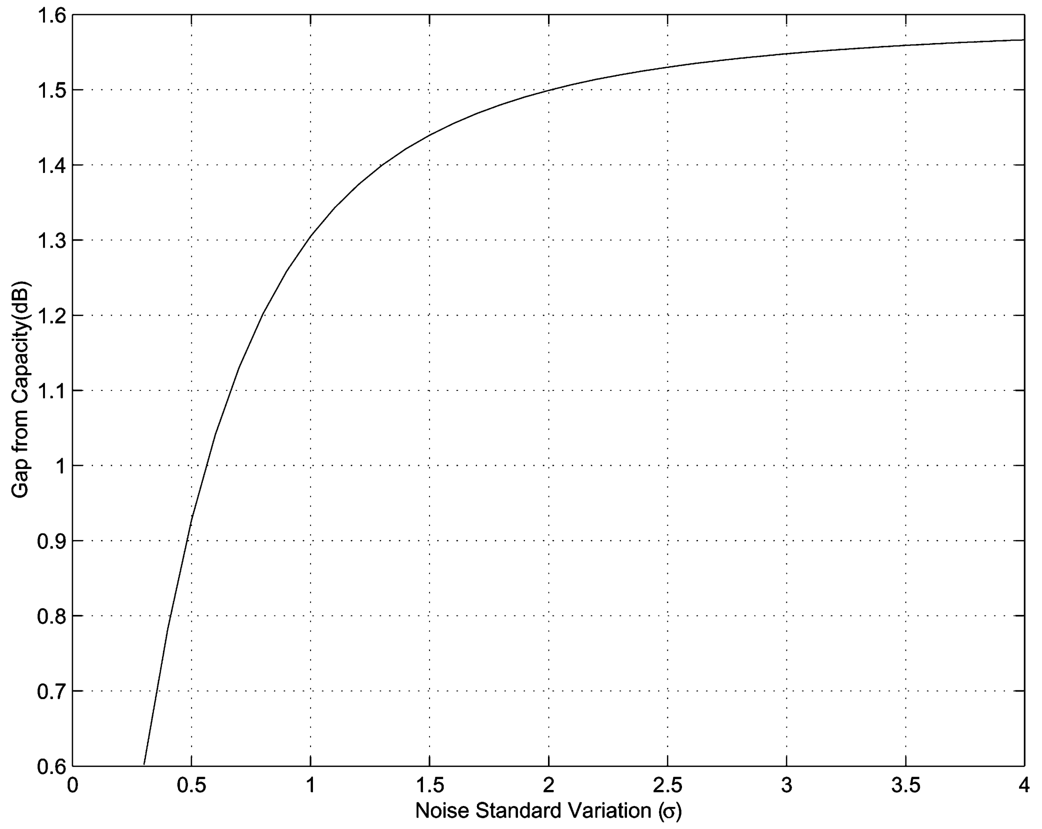

Fig. 6. The gap from the capacity for an ensemble of punctured LDPC codes over BIAWGN channel.

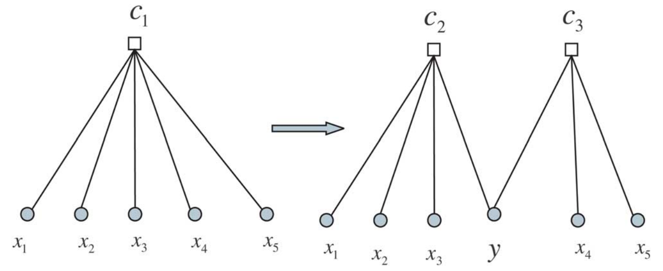

Fig. 7. Splitting a parity check equation.

By splitting the above parity-check equation, we get

$$\begin{aligned} c_2 &: x_1 \oplus x_2 \oplus x_3 \oplus y = 0 \\ c_3 &: x_4 \oplus x_5 \oplus y = 0 \end{aligned} \tag{22}$$

where we refer to y as an augmented variable node. Fig. 7 shows the effect of the parity-check splitting on the Tanner graph of the code. Note that this is different from the splitting operation introduced in [20]. Now consider an LDPC code in which some of the parity-check nodes have been split. This code can be considered as a punctured code. For example, the variable node y is a punctured variable node in Fig. 7. By splitting, we can make a graph correspond to a lower-rate code. When we puncture this code, we get a code with the same rate as the original code. Note that the splitting can be performed repeatedly and a check node that is obtained by splitting can itself be split into more check nodes. Therefore, we can have arbitrarily small rates. However, we assume that any check node is split only a finite number of times.

Consider an ensemble (λ,ρ) of LDPC codes of rate R_1. Let (λ,ρ,R_2) be the ensemble of codes obtained by splitting some of the check nodes of a code from the ensemble (λ,ρ) such that the rate of the code is decreased to R_2. Note that R_2 is the rate of the unpunctured codes. Obviously, puncturing the augmented variable nodes from a code in the ensemble (λ,ρ,R_2) results a code of rate R_1. For a graph g in the ensemble (λ,ρ), we define the graph g_{sp} to be the corresponding graph in (λ,ρ,R_2). For a check node c in a graph from the ensemble (λ,ρ), we define $Sp(c)$ as follows. If c is not split in g_{sp} then $Sp(c)=c$. Otherwise $Sp(c)$ is defined to be the set of check nodes in g_{sp} obtained by splitting c. For instance, in the above example we have $Sp(c_1)=\{c_2,c_3\}$. Similarly, for a set of check nodes A, we define

$$Sp(A)=\bigcup_{c\in A} Sp(c). \tag{23}$$

Lemma 3: If we puncture all the augmented bits from the ensemble (λ,ρ,R_2), then the resulting ensemble has the same threshold as the ensemble (λ,ρ) under message passing decoding algorithm.

Proof: Let v be a variable node in a graph g from the ensemble (λ,ρ). Let also $P_v^{(l)}$ be the probability that the estimate of the variable node v in the l'th iteration of the message passing algorithm be wrong. Define $P_v{}'(l)$ to be the corresponding probability when the decoding is performed on g_{sp}. Let $C_v^{(l)}$ and $V_v^{(l)}$ be respectively the sets of check nodes and variable nodes in the neighborhood of v that affect $P_v^{(l)}$. Define the sets $C_v'^{(l)}$ and $V_v'^{(l)}$ accordingly.

As it is shown in [12], with high probability the neighborhood of v (of constant depth) is tree-like. Therefore, $P_v^{(l)}$ is equal to the error probability of the ML estimation of v given the check nodes in $C_v^{(l)}$ and initial LLRs of the variable nodes in $V_v^{(l)}$. It is easy to show that the neighborhood of v in g_{sp} is tree-like as well. Choose $l' < \infty$ large enough such that $Sp\left(C_v^{(l)}\right) \subseteq C'(l')_v$ and $V_v^{(l)} \subseteq V'(l')_v$. Therefore, $P_v{}'(l')$ is equivalent to the error probability of the ML-estimator of v given more information than what is provided by $C_v^{(l)}$ and $V_v^{(l)}$. Hence, $P_v{}'(l') \le P_v^{(l)}$. Thus, if the average error probability under the message passing decoding on the graph g tends to zero as l goes to infinity, the same thing should be true for g_{sp}. Conversely, it can be shown that if the average error probability under the message passing decoding on the graph g_{sp} tends to zero as l goes to infinity, the same thing should happen for g. Thus, we conclude that the thresholds of the two ensembles are the same. □

An immediate result of Lemma 3 is the following theorem.

Theorem 5: For any ensemble (λ,ρ) of LDPC codes of rate $R_1>0$, and any number R_2 satisfying $R_2<R_1<1$, there exists an ensemble of punctured LDPC code of rate R_1 with parent rate R_2 having the same threshold under the belief propagation algorithm.

Theorem 5 implies that if we design the codes properly, punctured codes are as good as ordinary LDPC codes. It also shows how to construct a punctured code with the same performance as the unpunctured code.

It is worth noting that although Lemma 3 states that the graphs g and g_{sp} have the same asymptotic thresholds, the two codes can have different finite-length performance. In fact, by a suitable choice of the punctured variable nodes, we may be able to alleviate the destructive effect of short cycles in the Tanner graph. When the code length is short, the short cycles of the graph deteriorate the performance. Using splitting, we can increase the cycle lengths and this can improve the performance of the finite-length codes.

E. Puncturing Over the Binary Erasure Channel

For the erasure channel stronger results can be obtained. For example, using only one encoder and decoder we can achieve the capacity of BEC on arbitrary set of rates. As another example, a stronger result than Lemma 3 is valid. Note that Lemma 3 holds only for the asymptotic threshold of the codes. We now show that for any graph g, the error probability of the codes corresponding to g and g_{sp} are the same even for finite-length codes. Let us define the merging of two check nodes that are connected to a punctured node of degree two as the reverse operation of splitting. For instance, in the above example if we replace the two check nodes c_2 and c_3 by c_1 and delete the vertex y from the graph, we say we have merged c_2 and c_3.

Lemma 4: Let g be a bipartite graph with bipartition $V(g)$ and $C(g)$, the set of variable nodes and check nodes, respectively. Let also g_{sp} be a bipartite graph obtained by splitting some check nodes in g. Define $V(g_{sp})$ as the set of variable nodes in g_{sp}. We write $V(g_{sp}) = V_{sp} \cup V_p$, where $V_{sp} = V(g)$ and V_P is the set of augmented variable nodes. A set $S \subseteq V(g)$ is a stopping set in g if and only if there exists a set $U \subseteq V_p$ such that $S \cup U$ is a stopping set in g_{sp}.

Proof: Let $S \subseteq V(g)$ be a stopping set in g. Let $N_{g_{sp}}(S)$ be the set of neighbors of S in g_{sp} (i.e., the set of parity-check nodes in g_{sp} that are connected to some variable nodes in S). If for any $c \in N_{g_{sp}}(S)$ we have $\deg_S(c) \geq 2$ then S is a stopping set in g_{sp}, as well. Otherwise, there exists a parity-check equation $c_1 \in N_{g_{\text{sp}}}(S)$ with $\deg_S(c_1) = 1$. Since S is a stopping set in g, c_1 must have been split from a check node c in g. Suppose c has been split to $c_1, c_2, \ldots, c_j$. Since S is a stopping set in g, at least one of the check nodes $c_2, \ldots, c_j$ has a neighbor in S. Suppose c_t is connected to the variable node w in S. Thus, there is a path $c_1 - v_1 - c_2 - v_2 - \cdots - v_{t-1} - c_t - w$ in which $v_1, v_2, \ldots, v_{t-1}$ are augmented nodes of degree two. Thus $S \cup \{v_1, v_2, \ldots, v_{t-1}\}$ is a stopping set in g_{sp}.

Now suppose the sets $S \subseteq V(g) = V_{\text{sp}}$ and $U \subseteq V_p$ be such that the set $S \cup U$ is a stopping set in g_{sp}. Let C_s be the set of check nodes in $N_{g_{\text{sp}}}(S)$ that have a neighbor in U. We merge these check nodes to the original check nodes in g. For instance, suppose we merge the check nodes $c_1, c_2, \ldots, c_j$ to get a new check node c. We now show that c has at least two neighbors in S. Let G' be the graph induced by the nodes in $S \cup U$ and their neighbors in g_{sp}. Since c had been split to the check nodes $c_1, c_2, \ldots, c_j$, there is a path $c_1 - v_1 - c_2 - v_2 - \cdots - v_{j-1} - c_j$ in G' where $v_1, v_2, \ldots, v_{j-1}$ are the augmented nodes of degree two. However, the degrees of c_i's are at least two in the graph G'. Therefore, both c_1 and c_j must have neighbors in S. Thus, the check node c must have at least two neighbors in S. Thus we conclude that S is a stopping set in g. □

The immediate result of Lemma 4 is the following theorem.

Theorem 6: Let g be a bipartite graph and g_{sp} be a bipartite graph obtained by splitting some check nodes in g. Now suppose we puncture all the augmented variable nodes from the graph g_{sp}. The resulting code has the same bit error probability as the code that corresponds to g over the erasure channel under standard iterative decoding. The bit error rate is assumed to be calculated for only unpunctured bits.

In the previous section, we stated a general result regarding puncturing in Theorem 3. Here, we show stronger results for the BEC. For example, random puncturing of a code over BEC results in no performance loss. Using this fact we can show that for any R, $0 < R < 1$, it is possible to design a code of rate R with the following property. The code is capacity achieving if it is randomly punctured to any rate $R_1 \geq R$.

Lemma 5: Let C_1 be an LDPC code of rate R_1 and length n. Let e_1 be the bit error rate of the standard iterative decoding of the code when used over a BEC with erasure probability ϵ_1. Consider the ensemble (C_1, p) of LDPC codes that is obtained by randomly puncturing the code C_1 with puncturing fraction p. Choose ϵ_2 such that

$$\frac{1-\epsilon_1}{R_1} = \frac{1-\epsilon_2}{R_2} \tag{24}$$

where $R_2 = \frac{R_1}{1-p}$ is the rate of the ensemble (C_1, p). Let e_2 be the average bit error rate of a randomly chosen code from the ensemble (C_1, p) over a BEC with erasure probability ϵ_2. Then we have

$$e_1 = e_2. \tag{25}$$

Proof: This lemma is a special case of [11, Th. 3]. □

It is also easy to show that the bit error rate of a randomly chosen code from the ensemble (C_1, p) is highly concentrated about the average value using the same arguments as in [7] and [12]. Now we can state the following theorem.

Theorem 7: Let $T \subseteq (0,1)$ with $\inf_{R \in T} R > 0$ and δ be any positive constant. Then, there exists an ensemble (λ, ρ) of LDPC codes with the following property. The ensemble (λ, ρ) can be punctured randomly to get an ensemble of the arbitrary rate $R \in T$ such that

$$\forall\, R \in T; R \geq (1-\delta)c_R. \tag{26}$$

Here, $c_R = 1 - \epsilon_{\text{th}}(R)$ and $\epsilon_{\text{th}}(R)$ is the threshold of the punctured ensemble of rate R under the standard iterative decoding.

Proof: Let $R_p = \inf_{R \in T} R$ and let $\{(\lambda_N, \rho_N)\}$ be a sequence of capacity achieving degree distributions of rate R_p[7], [16], [17] and [18]. Choose N large enough such that $R_p \geq (1-\delta)c_{R_P}$. Since

$$\frac{c_{R_p}}{R_p} = \frac{c_R}{R}$$

for all $R \in T$, by Lemma 5 we conclude the proof. □

In other words, when we have a capacity achieving sequence of LDPC codes of rate R, the ensemble remains capacity achieving when it is punctured to a higher rate. Thus, we can design only one encoder and decoder and obtain arbitrary many rates. Moreover, the code is capacity achieving for all the desired rates over the BEC.

F. Design of Good Punctured LDPC Codes

In this section we discuss the design of good rate-compatible LDPC codes using puncturing. As it was shown in the previous section, design of punctured LDPC codes over the BEC is very simple. We just need to use a good degree distribution for the parent code and the punctured code performs very well for all higher rates. On the other hand, it is not obvious how to design good puncturing schemes for other channels.

We first note that if the desired range of rates is short, then we can choose a good code for the smallest rate and randomly puncture it to get codes of higher rates. Simulations shows that this simple structure is practically efficient. Our experience shows that increasing the code rate by an amount less than forty percent is usually obtained by a very small performance loss. As an example, the diagrams in [5], suggest that when the rate increase is less than forty percent, the performance loss is less than 0.2 dB for the BIAWGN channel. This is true even when we are using random puncturing. Thus we may focus on the cases that a broad range of rates is needed, specifically when rates close to 1 are needed.

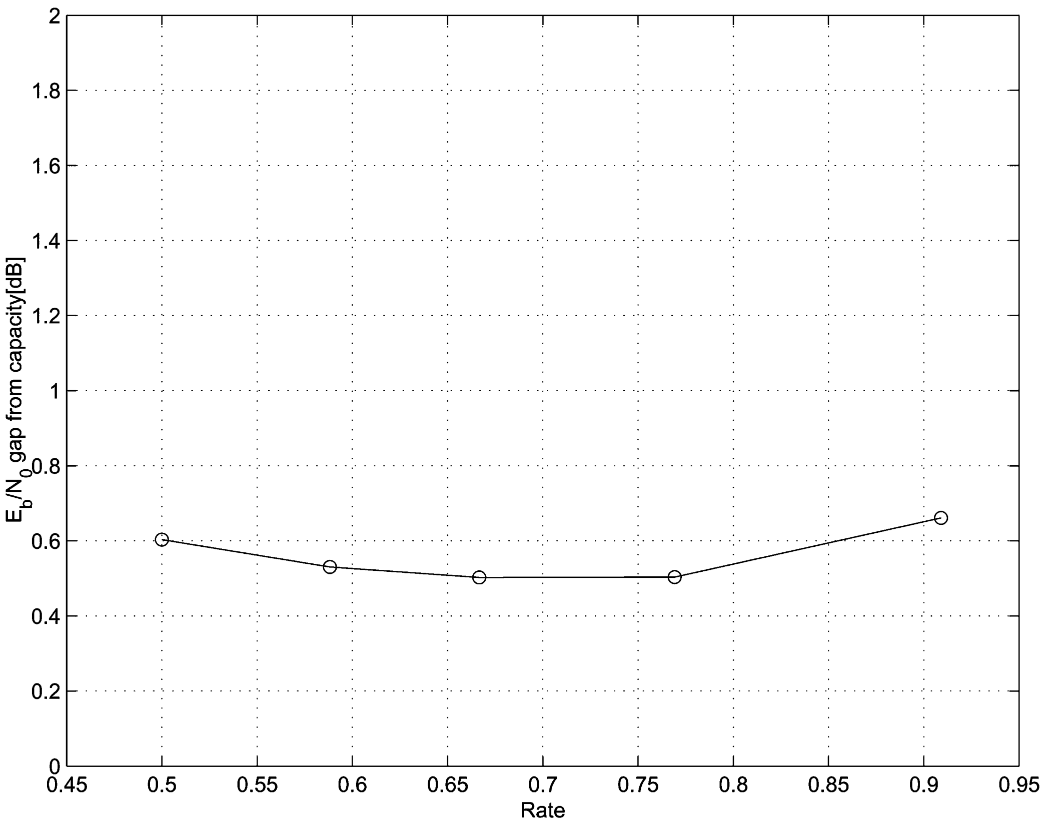

Fig. 8. The gap from capacity for a randomly punctured LDPC code of length 10^5 chosen from the ensemble (λ_5, ρ_5) at the bit error rate of 10^{-4}.

A necessary condition is to choose a code with high enough puncturing threshold. That is, if R_m and R_p are the highest desired rate and the parent code rate, respectively, and p_{th} is the puncturing threshold of the parent code, we must have $p_{\text{th}} > 1 - \frac{R_p}{R_m}$. Here we propose a technique that works well in practice.

As we mentioned previously a punctured code can be viewed as Fig. 4. Hence, we can consider the puncturing process as a variation in the channel not a variation in the code rate. From the previous section we know that the highest rate that we need plays an important role in the performance of the punctured code. When the puncturing fraction is maximum, the channel is close to a BEC. For the random puncturing, the resulting binary erasure channel is assumed to be uniform but for the intentional puncturing, the BEC is assumed to be nonuniform. A simple method is to choose the parent code to be a good code for BEC. By this choice, we expect to get good performance at the highest rates. However, as it is discussed in [21], with a little care, the code that is optimized over BEC is also optimal over other BIOSM channels. Thus we expect to get good performance even at very low rates. In fact, our experience shows that the most destructive problem of the punctured codes is the threshold effect. If the gap between the highest rate and the cut off rate is not enough, large performance degradation occurs at high rates. In order to examine the above methodology we chose a good ensemble of half-rate LDPC codes in [22] with the following degree distribution:

$$\begin{aligned}\lambda_5(x) &= 0.2498x + 0.2472x^2 \\ &\quad + 0.1480x^5 + 0.0033x^6 + 0.3517x^{19} \\ \rho_5(x) &= x^7. \end{aligned} \tag{27}$$

The ensemble has the cut off rate of $R_c = 0.9797$. We generate an LDPC code of length 10^5 from this ensemble. To compare our results with [5], we measured the gap to Shannon limit of this code at the bit error rate of 10^{-4}. Fig. 8 shows the simulation results for this code when it is randomly punctured to generate the rates at the range of 0.5 to 0.91. We note that for all rates the gap from the capacity is less than 0.7 dB. To compare the performance of this code to the codes given in [5], we examine the performance of two half-rates codes in [5] that are punctured to higher rates. We note that both codes show about 1.8 dB gap to the capacity at rate 0.91. Even with optimized intentional puncturing the codes have 1 dB gap to capacity at this rate. We also note that our code has a smaller length than those in [5].

It is worth noting that random puncturing is more suitable than intentional puncturing for rate-compatible coding. This is because we choose a fraction p_1 of the variable nodes at random for the first rate. For the next rate, we choose more bits at random from the unpunctured bits and so on. Thus, we do not need optimization for puncturing, reduce the degradation at higher rates, and do the puncturing in a rate-compatible way.

An important property of the above scheme is that it is extendable to finite-length codes. Using recent breakthrough in the design and analysis of finite-length LDPC codes over the BEC [1], [23], and [24] we can find good LDPC codes over the BEC and design efficient punctured LDPC codes.

VI. Capacity Achieving Sequences for BIOSM Channels Using Punctured Codes

It has been conjectured that for any BIOSM channel there exists a sequence of capacity achieving LDPC codes with iterative decoding, see for example [6]. Although this has been one of the most important open problems in coding theory, it has been proven only for the BEC. In this section we show that punctured LDPC codes may be helpful to verify the conjecture. We show that if the conjecture is proved for the rates approaching zero, then it will be valid for all rates.

Consider the class of BIOSM channels parameterized by a parameter θ with capacity c_θ. If the threshold of an ensemble of codes under the message passing decoding is $\theta_{\rm th}$, we say that the capacity of the ensemble is $c_{\theta_{\rm th}}$. We say that a sequence of degree distributions $\{(\lambda_n,\rho_n)\}_{n=1}^{\infty}$ with rate R is capacity achieving if for any positive constant δ, there exists an integer N such that if $n > N$, then $R \geq (1-\delta)c_{\theta_{{\rm th}_n}}$. Here $c_{\theta_{{\rm th}_n}}$ is the capacity of the ensemble (λ_n,ρ_n). We now suggest the following research problem.

Research Problem 1: For any class of BIOSM channels, there exists a sequence of degree distributions $\{(\lambda_n(R),\rho_n(R))\}_{n=1}^{\infty}$ such that

$$\lim_{R\to 0}\frac{R}{c(R)} = 1 \tag{28}$$

where $c(R) = \liminf_{n\to\infty} C_n(R)$, and $C_n(R)$ is the capacity of the ensemble $(\lambda_n(R),\rho_n(R))$.

We now show that if the above research problem is proved, then capacity achieving LDPC codes of all rates exist. The important point is that it might be easier to prove the existence of the capacity achieving sequences for the case where the rate of the code is approaching zero. This is because, for this case, the channel is approaching a channel with zero capacity and all different BIOSM channels may become somewhat similar. For example, one approach to solving the research problem could be using the capacity achieving sequences for the BEC as their rates approach zero.

Theorem 8: If the statement of Research Problem 1 is true, then there exists a capacity achieving sequence of LDPC codes for any BIOSM channel.

Proof: Let C_θ be a BIOSM channel with capacity c_θ and let δ be any positive constant. Consider the channel $C_{\rm eq}$ in Fig. 4. The capacity of $C_{\rm eq}$ is equal to $c_{\rm eq}(p) = (1-p)c_\theta$, where p is the puncturing fraction. Suppose we are using random puncturing. The channel $C_{\rm eq}(p)$ can be considered as a BIOSM channel with the parameter p. When p approaches one the capacity of $C_{\rm eq}(p)$ tends to zero. Thus, by Research Problem 1, there exists a sequence of degree distributions $\{(\lambda_n(R),\rho_n(R))\}_{n=1}^{\infty}$ such that

$$\lim_{p\to 1}\frac{R}{c_{\rm eq}(p)} = 1. \tag{29}$$

This code can be used for reliable communication over $C_{\rm eq}(p)$. Thus for large enough n and p, we have an ensemble of LDPC codes of rate $R \geq (1-\delta)c_{\rm eq}(p)$. This ensemble can be considered as a punctured ensemble with effective rate $R_{\rm eff} = R/(1-p)$. Therefore, we have

$$R_{\rm eff} = \frac{R}{1-p} \geq \frac{(1-\delta)c_{\rm eq}(p)}{1-p} = (1-\delta)c_\theta. \tag{30}$$

This implies that the punctured code with rate $R_{\rm eff} \geq (1-\delta)c_\theta$ can be used for reliable communication over C_θ. □

Note that Theorem 8 is quite general and can be applied to any code ensemble. In fact, the proof does not require to consider LDPC codes.

VII. Conclusion

We first presented improved decoding algorithms for LDPC codes over BIOSM channels. The algorithms have a considerably smaller bit error rate than the standard iterative decoding algorithm. For the BEC, both the average running time and the maximum running time of the proposed algorithm (Algorithm C) are small. For other BIOSM channels, the average running time of the proposed algorithm (Algorithm D) is almost the same as the standard iterative decoder. However, the maximum running time of the algorithm can be as large as 40 times the average running time of the iterative decoder. We showed that if the algorithm applied properly, it can be more effective on nonuniform channels.

We then studied some fundamental properties of punctured LDPC codes. The threshold effect and the optimality of punctured LDPC codes were discussed. We showed that for any ensemble of LDPC codes, there exists a cut off rate which is the maximum achievable rate using puncturing. We proved the existence of asymptotically good punctured LDPC codes for an arbitrary range of rates. For the binary erasure channel, we find that using only one encoder and decoder, we can achieve the capacity of the channel over an arbitrary set of rates. We also proposed a simple method for designing rate-compatible LDPC codes that has several advantages over the previous methods. First, it reduces the performance degradation at high rates. Second, it is applicable to finite-length codes. Third, there is no need for optimizing the puncturing pattern. Fourth, the puncturing can be done in a rate-compatible way. Finally, we showed that punctured LDPC codes might be useful in solving an important open research problem on the capacity-achieving LDPC codes over BIOSM channels.

Appendix
Proof of Theorem 2

For simplicity we prove the theorem for random puncturing. The random puncturing scheme can be modeled as transmitting over a channel $C_{\rm eq}$ as shown in Fig. 4 with the following description. Assume a bit is transmitted through $C_{\rm eq}$. Then, with probability p, this bit will be transmitted over a channel with zero capacity and with probability $1-p$ it will be transmitted over the channel C_θ. Now, using similar discussion to the proof of Theorem 1, we can show that if in the above model we replace C_θ with the channel $C_{\theta_{\min}}$, which is a channel of capacity one, the overall error probability decreases. On the other hand, replacing C_θ with the channel $C_{\theta_{\min}}$ in Fig. 4, the channel $C_{\rm eq}$ becomes equivalent to a BEC with erasure probability $\epsilon = p$. This is because every bit is either transmitted through the channel with zero capacity C_2 (with probability p) or through the noiseless channel $C_{\theta_{\min}}$. This proves a lower bound on the error probability that results in the upper bound on the puncturing threshold.

Examining the above discussion indicates that the theorem applies when the error rate is averaged over all bits, i.e., both punctured and unpunctured bits. A more realistic case is to consider the error rate of only unpunctured bits. By a similar argument to [11, Lemma 1], for $\theta > \theta_{\min}$, if the error rate of punctured bits is bounded away from zero, then the error rate of unpunctured bits is bounded away from zero as well. This implies the following corollary.

Corollary 2: Consider the ensemble of LDPC codes defined by the pair (λ, ρ). Let (λ, ρ, p) be the ensemble of LDPC codes that are generated by randomly puncturing of the ensemble (λ, ρ) by the puncturing fraction p. Assume a randomly chosen code from the ensemble (λ, ρ, p) is used over the channel C_θ, with $\theta > \theta_{\min}$. Let ϵ_{th} be the threshold of the ensemble (λ, ρ) for BEC under the iterative decoding. If $p > \epsilon_{\text{th}}$, then the error probability of decoding the punctured code is bounded away from zero independent of the communication channel.

The above discussion gives upper bounds on the puncturing fraction of LDPC codes. We will now show that these upper bounds are actually the puncturing threshold of LDPC codes in the sense that if the puncturing fraction is less than the upper bounds then the punctured LDPC code is an asymptotically good code. We first prove a lemma.

Lemma 6: Consider the ensemble of LDPC codes defined by the pair (λ, ρ) that are randomly punctured by the puncturing fraction $p < p_{\text{th}}$ where $p_{\text{th}} = \epsilon_{\text{th}}$ is the upper bound given by Corollary 2. Then there exists a $\theta_1 > \theta_{\min}$ such that if the channel parameter θ is smaller than θ_1, then the punctured ensemble satisfies the stability condition.

Proof: By the assumption of the lemma, a randomly chosen code from the ensemble (λ, ρ) can be used to obtain arbitrarily small bit error rate over a BEC with erasure probability $\epsilon = p$. Thus, using the stability condition [6], we have

$$\lambda_2 \rho'(1) < \frac{1}{p}. \tag{31}$$

Now consider the ensemble of LDPC codes defined by (λ, ρ) that are randomly punctured by a puncturing fraction p. Suppose a code from the punctured ensemble is used over a channel with the parameter θ. Then, assuming the all-one codeword has been sent, the density of the LLRs from the channel is equal to

$$f_0(x) = p\delta(x) + (1-p) f_{z_\theta}(x). \tag{32}$$

We need to show that for suitably chosen $\theta > \theta_{\min}$ we have

$$\lambda_2 \rho'(1) < e^r, \quad r = -\ln\left(\int_{\mathbb{R}} f_0(x) e^{-\frac{x}{2}} dx\right). \tag{33}$$

Since as θ goes to $\theta_{\min}$, $f_{z_\theta}(x)$ converges to $\Delta_\infty(x)$ (in the sense defined in [6]). By choosing $\theta > \theta_{\min}$ small we can make the integral $\int_{\mathbb{R}} (1-p) f_{z_\theta}(x) e^{-\frac{x}{2}} dx$ arbitrarily small. Thus, using (31) and (32) we conclude that there exists a $\theta > \theta_{\min}$ for which we have $\lambda_2 \rho'(1) < e^r$ for $r = -\ln\left(\int_{\mathbb{R}} f_0(x) e^{-\frac{x}{2}} dx\right)$. □

Now we show under the conditions of Lemma 6, there exists a $\theta^* > \theta_{\min}$ such that if the channel parameter θ is smaller than θ^*, then

$$\lim_{l \to \infty} P_e(F_l) = 0. \tag{34}$$

For the given ensemble, the probability density function f_l can be written as

$$f_l(x) = y_l \delta(x) + (1 - y_l) g_l(x) \tag{35}$$

where y_l satisfies

$$\begin{aligned} y_0 &= p \\ y_l &= p\lambda(1 - \rho(1 - y_{l-1})). \end{aligned} \tag{36}$$

By the conditions of the theorem

$$\lim_{l \to \infty} y_l = 0. \tag{37}$$

Moreover, for a fix value of l, we have

$$\lim_{\theta \to \theta_{\min}} P_e(g_l) = 0. \tag{38}$$

Now, by Lemma 6, the stability condition is satisfied, for $\theta < \theta_1$. Thus, by the stability theorem in [6], there exists a constant $\zeta > 0$ such that if $P_e(F_l) < \zeta$, for some $l \in \mathbb{N}$, then $P_e(F_l)$ converges to zero as l tends to infinity. Choose l_1 large enough such that $y_{l_1} < \zeta$. Now fix l_1 and choose $\theta_2 > \theta_{\min}$ such that $P_e(g_{l_1}) < \zeta/2$. Thus for $\theta < min(\theta_1, \theta_2)$, the stability condition is satisfied and we have

$$P_e(F_{l_1}) = \frac{1}{2} y_{l_1} + (1 - y_{l_1}) P_e(g_{l_1}) < \zeta. \tag{39}$$

Therefore, $P_e(F_l)$ converges to zero as l tends to infinity.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful comments and suggestions. The authors are also grateful to Rudiger Urbanke, the Associate Editor, for handling the review.

REFERENCES

[1] C. Di, D. Proietti, I. E. Telatar, T. Richardson, and R. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 48, pp. 1570–1579, 2002.
[2] Y. Mao and A. Banihashemi, "A new schedule for decoding low-density parity-check codes," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, 2001, vol. 2, pp. 1007–1010.
[3] A. Nouh and A. Banihashemi, "Bootstrap decoding of low-density parity-check codes," *IEEE Commun. Lett.*, vol. 6, pp. 391–393, 2002.
[4] H. Pishro-Nik and F. Fekri, "On decoding of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 50, pp. 439–454, 2004.
[5] J. Ha and S. McLaughlin, "Optimal puncturing of low-density parity-check codes," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2003.
[6] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 619–637, 2001.
[7] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 569–584, 2001.

[8] C. Di, T. Richardson, and R. Urbanke, "Weight distributions: How deviant can you be?," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2001.

[9] A. Orlitsky, K. Viswanathan, and J. Zhang, "Stopping set distribution of LDPC code ensembles," *IEEE Trans. Inf. Theory*, vol. 51, pp. 929–953, 2005.

[10] H. Pishro-Nik, N. Rahnavard, and F. Fekri, "Nonuniform error correction using low-density parity check codes," in *Proc. 40th Annu. Allerton Conf.*, Urbana-Champaign, IL, Oct. 2002.

[11] ——, "Non-uniform error correction using low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp. 2702–2714, Jul. 2005.

[12] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, pp. 599–618, 2001.

[13] G. D. Forney, R. Koetter, F. Kschischang, and A. Reznik, "On effective weights of pseudocodewords for codes defined on graphs with cycles," *Proc. IMA*, 1999.

[14] B. J. Frey, R. Koetter, and A. Vardy, "Signal-space characterization of iterative decoding," *IEEE Trans. Inf. Theory*, vol. 47, pp. 766–781, 2001.

[15] R. Koetter and P. O. Vontobel, "Graph-covers and iterative decoding of finite length codes," in *Proc. 3rd Int. Symp. Turbo Codes and Rel. Topics*, Sep. 2003.

[16] M. A. Shokrollahi, "New sequences of linear time erasure codes approaching the channel capacity," in *Proc. 13th Int. Symp. Appl. Algebra, Algebraic Algorithms and Error-Correcting Codes*, 1999, pp. 65–76.

[17] ——, "Capacity-achieving sequences," *IMA Volumes in Mathematics and Its Applications*, vol. 123, pp. 153–166, 2000.

[18] P. Oswald and M. A. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Trans. Inf. Theory*, vol. 48, pp. 3017–3028, 2002.

[19] A. Khandekar, "Graph-Based Codes and Iterative Decoding," Ph.D. dissertation, California Inst. Technol., Pasadena, CA, 2002.

[20] 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, 2001.

[21] S. Y. Chung, "On the Construction of Some Capacity-Approaching Coding Schemes," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, 2000.

[22] [Online]. Available: http://lthcwww.epfl.ch/research/ldpcopt/

[23] T. Richardson, A. Shokrollahi, and R. Urbanke, "Finite-length analysis of low-density parity-check ensembles for the binary erasure channel," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2002.

[24] T. J. Richardson and R. L. Urbanke, Finite-Length Density Evolution and the Distribution of the Number of Iterations for the Binary Erasure Channel [Online]. Available: http://lthcwww.epfl.ch/research/ldpcopt/