Design of Irregular LDPC Codes with Optimized Performance-Complexity Tradeoff

Benjamin Smith, Masoud Ardakani, *Senior Member, IEEE*,
Wei Yu, *Senior Member, IEEE*, and Frank R. Kschischang, *Fellow, IEEE*

***Abstract*—The optimal performance-complexity tradeoff for error-correcting codes at rates strictly below the Shannon limit is a central question in coding theory. This paper proposes a numerical approach for the minimization of decoding complexity for long-block-length irregular low-density parity-check (LDPC) codes. The proposed design methodology is applicable to any binary-input memoryless symmetric channel and any iterative message-passing decoding algorithm with a parallel-update schedule. A key feature of the proposed optimization method is a new complexity measure that incorporates both the number of operations required to carry out a single decoding iteration and the number of iterations required for convergence. This paper shows that the proposed complexity measure can be accurately estimated from a density-evolution and extrinsic-information transfer chart analysis of the code. A sufficient condition is presented for convexity of the complexity measure in the variable edge-degree distribution; when it is not satisfied, numerical experiments nevertheless suggest that the local minimum is unique. The results presented herein show that when the decoding complexity is constrained, the complexity-optimized codes significantly outperform threshold-optimized codes at long block lengths, within the ensemble of irregular codes.**

***Index Terms*—Convex optimization, extrinsic-information transfer (EXIT) charts, decoding complexity, low-density parity-check (LDPC) codes.**

I. Introduction

THE design of high-performance irregular low-density parity-check (LDPC) codes has been studied extensively in the literature (e.g., [1]–[3]). At long block lengths, codes derived from capacity-approaching degree distributions (so-called threshold-optimized codes) provide excellent performance, albeit with high decoding complexity. In particular, to achieve the full potential of a threshold-optimized code, an impractically large number of messages may need to be passed in the iterative decoder. In this paper, we present a design method to find irregular LDPC codes with an optimized performance-complexity tradeoff, thus permitting excellent performance with reduced decoding complexity.

In designing threshold-optimized codes, decoding complexity is not explicitly considered in the code design process. However, if one wishes to design a practically decodable code, it would be more natural to try to find the highest-rate code for a certain affordable level of complexity on a given channel, or the code with the minimum decoding complexity for a required rate and a given channel condition, or the code with the highest decoding threshold at a given rate and complexity. Clearly, these design objectives better reflect the requirements of practical communications systems.

A main challenge in incorporating complexity in code design is that characterizing decoding complexity as a function of code parameters may appear, at a first glance, to be a difficult task. This is especially true for iterative decoding systems in which decoding complexity depends not only on the complexity of each iteration, but also on the number of iterations required for convergence. The main idea of this paper is that by using a uni-parametric representation of the decoding trajectory [4], [5], the required number of iterations in the iterative message-passing decoding of a long-block-length LDPC code under the parallel-update schedule can be accurately *estimated*. This allows one to define and to quantify a measure of decoding complexity, which can then be used to optimize the complexity-rate tradeoff for LDPC codes.

This paper uses a density evolution analysis of the LDPC decoding process and adopts a uniparametric representation of the decoding trajectory in terms of the evolution of message-error rate, in a format similar to an extrinsic-information transfer (EXIT) chart [4], [6], [7], except that message-error rate is tracked instead of mutual information, and that no Gaussian assumption is made. The use of probability-of-error-based EXIT charts (which originated from the work of Gallager [8]; see also [5]) offers two key advantages. First, the uniparametric representation of the decoding trajectory allows one to accurately estimate the number of iterations required to reduce the bit-error rate (BER) from that given by the channel to a desired target. Second, in terms of message error rate, one can show that the decoding trajectory of an irregular LDPC code can be expressed as a linear combination of trajectories of *elementary* codes parameterized by their variable degrees. These two facts allow one to define a measure that relates the decoding complexity of an irregular LDPC code to its degree distribution. The code design problem then reduces to the shaping of the decoding trajectory for an optimal complexity-

Paper approved by F. Fekri, the Editor for LDPC Codes and Applications of the IEEE Communications Society. Manuscript received April 14, 2008; revised February 12, 2009.

B. Smith, W. Yu, and F. R. Kschischang are with the Electrical and Computer Engineering Department, University of Toronto, 10 King's College Road, Toronto, Ontario M5S 3G4, Canada (e-mail: {ben, weiyu, frank}@comm.utoronto.ca).

M. Ardakani is with the Electrical and Computer Engineering Department, University of Alberta (e-mail: ardakani@ece.ualberta.ca).

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

The material in this paper was presented in part at the IEEE International Symposium on Information Theory, Adelaide, Sept. 2005 and at the 43rd Annual Allerton Conference on Communication, Control, and Computing, Allerton, IL, Sept. 2005.

Digital Object Identifier 10.1109/TCOMM.2010.02.080193

rate tradeoff. An optimization program that determines the minimum-complexity degree distributions for a fixed code rate and a target channel can then be formulated.

This paper shows that the optimization problem formulated in this way is convex under a mild condition, which facilitates its numerical solution. Numerical code design examples suggest that complexity-optimized codes can significantly outperform irregular threshold-optimized codes at long block lengths.

The methodology proposed in this paper offers one of the few instances of an iterative system in which analytic tools are available not only for predicting the convergence trajectory, but also for the tuning of system parameters for fast convergence. The proposed code design method is applicable to any binary-input memoryless symmetric (BMS) channel and any iterative decoding algorithm with symmetric update rules. These are the conditions required for the validity of density evolution analysis [2].

A. Related Work

The performance-complexity tradeoff for error-correcting codes has always been a central issue in coding theory [9]; in the context of capacity-approaching codes, there exist several information-theoretic results. For LDPC codes and irregular repeat-accumulate (IRA) codes, Khandekar and McEliece [10] conjectured that for symmetric channels with binary inputs, the graphical complexity (i.e., the decoding complexity per iteration) per information bit scales like $\log \frac{1}{\varepsilon}$, and the number of iterations scales like $\frac{1}{\varepsilon}$, where ε is the multiplicative gap to capacity. In the special case of the binary erasure channel (BEC), Sason and Wiechman [11] proved that the number of iterations scales at least like $\frac{1}{\varepsilon}$ for LDPC, systematic IRA, non-systematic IRA, and systematic accumulate-repeat-accumulate (ARA) codes. Furthermore, the graphical complexity per information bit of LDPC and systematic IRA codes has been shown to scale like $\log \frac{1}{\varepsilon}$ [12], [13], while non-systematic IRA and systematic ARA codes can achieve bounded graphical complexity per information bit [14], [15]; in practice, systematic ARA are preferable to non-systematic IRA codes due to their lower error floors and systematic encoding.

With respect to more practically-oriented studies of the performance-complexity tradeoff, Richardson et al. [2] proposed a numerical design procedure for irregular LDPC codes involving an ad-hoc measure of the number of iterations. However, their motivation was in providing an efficient technique for finding threshold-optimized codes. A decoding complexity measure similar to ours is presented in [16], however, its usefulness is limited to capacity-approaching codes over the BEC; ours is more general, but reduces to their measure in the case of capacity-approaching codes over the BEC. In a related paper [17], we studied the optimization of decoding complexity for LDPC codes under Gallager's decoding Algorithm B [8] over the binary symmetric channel. The results of [17] rely on the one-dimensional nature of decoding Algorithm B; this paper addresses the more general case.

B. Outline of the Paper

The rest of this paper is organized as follows. In Section II, we briefly review the background and terminology pertaining to LDPC codes. In Section III, we present the new complexity measure for the iterative decoding system, and formulate the complexity optimization methodology. Numerically optimized degree distributions and simulation results are presented in Section IV, and conclusions are provided in Section V.

II. Preliminaries

A. LDPC Codes

In this paper we consider the ensemble of irregular LDPC codes [1], [2], characterized by a variable degree distribution $\lambda(x)$ and a check degree distribution $\rho(x)$,

$$\lambda(x) = \sum_{i \geq 2} \lambda_i x^{i-1} \text{ and } \rho(x) = \sum_{j \geq 2} \rho_j x^{j-1}$$

with $\lambda(1) = \rho(1) = 1$. The rate of an LDPC code is related to its degree distribution by

$$R = 1 - \frac{\sum_j \frac{\rho_j}{j}}{\sum_i \frac{\lambda_i}{i}}. \tag{1}$$

Due to their representation via sparse bipartite graphs, LDPC codes are amenable to iterative decoding techniques. Iterative decoding proceeds by successively passing messages between variable nodes and check nodes, where the messages represent 'beliefs'—typically expressed as log-likelihood ratios (LLRs)—about the values of the variable node connected to a given edge. In a single decoding iteration, messages $m_{v \to c}$ are first sent (in parallel) from variable nodes to check nodes. At each check node a check-update computation is performed, generating messages $m_{c \to v}$ to be sent (in parallel) from the check nodes to the variable nodes. Finally, at each variable node, a variable-update computation is performed to generate the messages for the next iteration. After sufficiently many iterations have been performed, each variable node can produce a decision about its corresponding variable. In the rest of this paper, we assume that the check- and variable-node updates are performed according to the sum-product algorithm [18] with a parallel message-passing schedule, although our technique is applicable to any symmetric update rules. Note that even though one may reduce the decoding complexity by employing simplified (sub-optimal) updates, our method provides further complexity reduction by minimizing the total number of messages passed.

For the sum-product algorithm, the update rule for an LLR message at a variable-node v is

$$m_{v \to c} = m_0 + \sum_{h \in n(v) - \{c\}} m_{h \to v}, \tag{2}$$

and the update rule at a check-node c is

$$m_{c \to v} = 2 \tanh^{-1} \left(\prod_{y \in n(c) - \{v\}} \tanh(m_{y \to c}/2) \right), \tag{3}$$

where m_0 is the channel message in LLR form, and $n(v)$ represents the nodes connected directly to node v by an edge.

In a pair of landmark papers [2], [19], Richardson et al. presented density evolution, a numerical tool to track the density of messages passed during iterative decoding of LDPC codes, under the parallel message-passing schedule. Given a degree distribution pair $(\lambda(x), \rho(x))$ and a target channel, one can use density evolution to determine the 'threshold' of the code, where the threshold corresponds to the largest value of the noise parameter such that the bit-error probability goes to zero, under iterative decoding, as the block length tends to infinity.

In one iteration of density evolution, the algorithm combines the probability density function (pdf) of channel messages, the pdf of messages sent from variable nodes to check nodes in the previous iteration, and the degree distributions of the code, and computes the pdf of the messages that will be sent from variable nodes to check nodes in the current iteration. In other words, one iteration of density evolution computes

$$\mathcal{D}_{v\to c}^{(k)} = \Phi(\mathcal{D}_{v\to c}^{(k-1)}, \mathcal{D}_{ch}, \lambda(x), \rho(x)), \tag{4}$$

where $\mathcal{D}_{v\to c}^{(k)}$ represents the pdf of variable-node to check-node messages at iteration k, and $\mathcal{D}_{ch}$ represents the pdf of channel messages.

B. EXIT Charts

EXIT charts, originally introduced by ten Brink in the context of turbo codes [4], provide a graphical description of density evolution. This is accomplished by tracking a statistic of the variable-to-check message density $\mathcal{D}_{v\to c}^{(k)}$. Often the mutual information between the message value and the corresponding variable value is tracked in an EXIT chart. In this paper, as in [5], [8], we track a different parameter of the message distribution; namely, the probability that the log-likelihood ratio (LLR) message has the incorrect sign. Roughly speaking, this parameter measures the fraction of messages in "error". Although we do not track mutual information, we nevertheless continue to refer to the parameterized trajectories as "EXIT charts." The use of EXIT charts is central to the formulation of a threshold-optimization program, and later, a complexity-minimization program. However, to minimize the loss of information associated with a uni-parametric representation of message densities, we use density evolution for analysis; the EXIT charts serve only to provide a graphical representation of the convergence trajectory.

For a given degree distribution pair and channel condition, one can visualize the convergence behavior of the decoder by performing N iterations of density evolution, and plotting the message error rate (i.e., the fraction of variable-to-check messages $m_{v\to c}$ with the incorrect sign) at iteration k, $k \in \{1, 2, \cdots, N\}$, denoted as $p^{(k)}$, versus the message error rate at iteration $k-1$, i.e., $p^{(k-1)}$. Specifically, let $\mathcal{D}_{ch}$ be the conditional pdf of the LLR at the output of the channel given that $x = 1$ is transmitted (we assume $0 \to +1$ and $1 \to -1$ for BPSK modulated signals), and let $\mathrm{P_e}(\cdot)$ be a function that returns the area under the error-tail of a pdf. Clearly,

$$p^{(0)} = \mathrm{P_e}(\mathcal{D}_{ch})$$

is the bit error probability associated with uncoded communications. For $k \geq 1$, we perform density evolution via equation (4), where $\mathcal{D}_{v\to c}^{(0)}$ is initialized to the Dirac delta function at zero, and we obtain

$$p^{(k)} = \mathrm{P_e}(\mathcal{D}_{v\to c}^{(k)}),$$

the message error at iteration k. Plotting $p^{(k)}$ versus $p^{(k-1)}$ gives the EXIT chart of the code for the assumed channel condition. Equivalently, we can think of the EXIT chart as a mapping

$$p^{(k)} = f(p^{k-1}),\ k \in \{1, 2, \ldots\}. \tag{5}$$

Due to the random construction of the graph of an irregular LDPC code, one can use the total probability theorem to decompose the overall EXIT chart into a sum of elementary EXIT charts [5], i.e.,

$$f(p) = \sum_i \lambda_i f_i(p, \lambda), \tag{6}$$

where $f_i(p, \lambda)$ is the elementary EXIT chart associated with degree-i variable nodes, and the check degree distribution is fixed to an appropriate value. For the calculation of the i-th elementary chart, we first compute

$$q_i^{(k)} \triangleq \mathrm{P_e}(\Phi(\mathcal{D}_{v\to c}^{(k-1)}, \mathcal{D}_{ch}, x^{i-1}, \rho(x))),\ k \in \{1, 2, \ldots\}$$

which is the error rate of the messages emanating from degree i variable nodes after the k-th iteration, where $\mathcal{D}_{v\to c}^{(k-1)}$ is the pdf of messages after $k-1$ iterations of density evolution on the code with degree distributions $\lambda(x)$ and $\rho(x)$. The i-th elementary EXIT function f_i is then obtained by interpolating a curve through the origin and the points $(p^{(k-1)}, q_i^{(k)})$, $k \geq 1$. Fig.1 presents the elementary EXIT charts and the overall EXIT chart for a particular irregular code, using sum-product decoding over the AWGN channel.

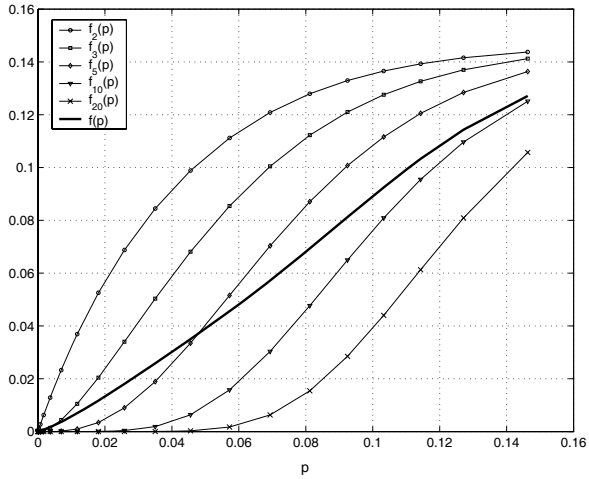


Fig. 1. Elementary EXIT charts, $\lambda(x) = 0.1x + 0.3x^2 + 0.1x^4 + 0.2x^9 + 0.3x^{19}$, $\rho(x) = x^7$.

Note that the elementary EXIT charts are in general a function of λ. This is because the densities that are passed in iteration m are influenced by λ. However, if λ is perturbed slightly, the elementary EXIT charts are effectively unchanged, and thus for sufficiently small changes in λ, we disregard

the dependency of elementary EXIT charts on λ.[1] With this assumption, (6) can be simplified to

$$f(p) = \sum_i \lambda_i f_i(p), \tag{7}$$

therefore, the code-design problem becomes that of synthesizing a suitable EXIT chart as a convex combination of a number of pre-computed elementary EXIT charts. Finally, due to the weak dependence of elementary EXIT charts on λ, the accuracy of the design method can be further improved by updating the elementary EXIT charts whenever λ undergoes a sufficiently large change.

C. Threshold Optimization

Traditionally, degree distributions have been designed to achieve a desired rate, while maximizing the decoding threshold. Equivalently, given a target channel condition, one can determine the maximum rate code such that successful decoding is possible. Indeed, for a fixed check degree, the latter is obtained as the solution to the following linear optimization program:

$$\begin{aligned} \text{maximize} \quad & \sum_i \frac{\lambda_i}{i} \\ \text{subject to} \quad & \sum_i \lambda_i f_i(p) < p, \quad p \in (0, \mathrm{P_e}(\mathcal{D}_{ch})] \\ & \sum_i \lambda_i = 1 \\ & \lambda_i \geq 0 \end{aligned} \tag{8}$$

Here the central condition is the first constraint, namely that $\sum_i \lambda_i f_i(p) < p$; this condition ensures an "open" EXIT chart in which the decoder can make progress (decrease p) during each iteration. One clearly observes the code-design problem as "curve-shaping."

III. Complexity-Optimized LDPC Codes

In this section, we formulate an optimization program that finds the minimum-complexity code as a function of code rate on a fixed channel. The key to this formulation is a measure that accurately estimates the number of computations needed to successfully decode an irregular LDPC code using the sum-product algorithm. A binary-input additive white Gaussian noise (AWGN) channel is assumed, although the proposed optimization technique applies to any binary-input memoryless symmetric channel.

A. Measure of Decoding Complexity

For a parallel message-passing decoder, the decoding complexity of LDPC codes is proportional to the product of the number of decoding iterations and the number of arithmetic operations performed per iteration. The computational effort (in computing (2) and (3)) per iteration is proportional to the number of edges E in the graph of the code, while the number of messages passed per iteration is easily seen to be $2E$. Thus, as we argue below, the overall complexity is proportional to the total number of messages passed in the iterative decoding process.

To see that the complexity per iteration scales linearly with E, we may explicitly compute (for the sum-product algorithm) the number of operations performed in one iteration; the analysis of other update rules is similar. The variable-node update for the sum-product algorithm (2) can be performed as follows. At each variable node of degree d_v, we first compute the sum of the d_v extrinsic messages and the channel message; this requires d_v additions. To compute the output message for some particular edge, we can then subtract the extrinsic message received over the same edge from the computed sum. Thus, computing the m output messages requires $2d_v$ operations. Summing over all variable nodes, $O(E)$ operations are required.

Similarly, to perform the check-update (3) at a check node of degree d, we can first compute $\tanh(\frac{m}{2})$ for each incoming message m. Next, performing $d-1$ operations, we compute the product of these terms. Finally, computing an output requires dividing this quantity by the term associated with the incoming message over the same edge, then taking $2\tanh^{-1}$ of the resulting value. Thus, computing all the check-node output messages for a check node of degree d requires $d-1$ multiplications, d divisions, and $2d$ $\tanh/\tanh^{-1}$ calculations. Summing over all check nodes, $O(E)$ operations are needed in total.

The overall complexity for decoding one codeword is therefore proportional to NE, where N is the number of decoding iterations. Since each codeword encodes Rn information bits, where R is the code rate and n is the block length, the decoding complexity per information bit is then $O\left(\frac{NE}{Rn}\right)$. Now, using the fact that $n = E\sum \frac{\lambda_i}{i}$ and using the expression (1), which relates R with λ_i and ρ_i, we obtain the following expression for the decoding complexity K

$$K = N\frac{E}{Rn} = N\frac{1}{R\sum \frac{\lambda_i}{i}} = N\frac{1-R}{R\sum \frac{\rho_i}{i}}. \tag{9}$$

In the rest of this section, we formulate the complexity-rate tradeoff problem as that of minimizing K for a fixed R. Furthermore, since the average check node degree can be shown to be $\bar{d}_c = \left(\sum_i \frac{\rho_i}{i}\right)^{-1}$, minimizing the decoding complexity becomes equivalent to minimizing $N\bar{d}_c$, and the search for complexity-minimized codes involves an investigation of the tradeoff between average check node degree and the number of decoding iterations.

B. Characterizing the Number of Decoding Iterations

The main idea of this paper is that the total number of decoding iterations can be accurately estimated based on the shape of the EXIT chart $f(p)$. Further, this characterization of complexity can be expressed as a differentiable function of code parameters (i.e., λ_i), which allows one to use a continuous optimization approach for finding good codes with reduced decoding complexities.

From the definition of EXIT charts, it is clear that computing the decoding trajectory for a given code is equivalent

[1]Intuitively, this is justified by the accuracy of methods that assume a Gaussian distribution for variable-node to check-node messages, which leads to elementary EXIT charts that are *independent* of λ; elementary EXIT charts computed in this manner closely approximate elementary EXIT charts computed by density evolution [5], [20].

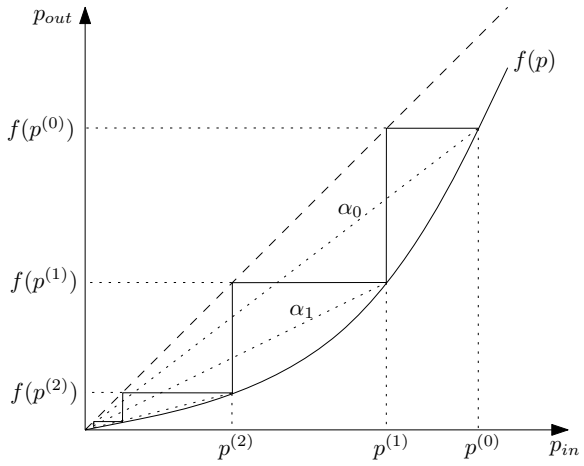


Fig. 2. The iterative decoding process can be represented by an iterative function evaluation of the EXIT chart, i.e, $p^{(k)} = f(p^{(k-1)})$. Further, $p^{(k)}$ and $p^{(k-1)}$ are related by the local slope $\alpha_k = \frac{p^{(k+1)}}{p^{(k)}}$.

to an iterative function evaluation of $f(p)$. Given some initial message error rate $p_0 > 0$ and a target message error rate $p_t < p_0$, let $p^{(0)} = p_0$. The iterative decoding process is represented by (5), i.e., $p^{(k)} = f(p^{(k-1)})$ for $k \in \mathbb{N}$. Convergence to p_t implies the existence of some $j \in \mathbb{N}$ such that $p^{(j)} = f(p^{(j-1)}) \leq p_t$. The required number of iterations for convergence, N, is defined to be the minimum such j. Note that the number of required iterations is determined by the shape of $f(p)$ alone.

Note that a sufficient condition for convergence to p_t is that $f(p) < p, \forall p \in [p_t, p_0]$, since this implies that the output of an iteration is strictly less than its input, and that the recursion has no fixed points in the interval of interest.

The following definition plays an important role in the characterization of the required number of iterations for convergence. Define the *local slope* of $f(\cdot)$ at $p^{(k)}$ to be

$$\alpha_k = \frac{f(p^{(k)})}{p^{(k)}}. \tag{10}$$

Since $p^{(1)} = f(p^{(0)})$, $p^{(2)} = f(p^{(1)})$, $\cdots$, $p^{(N)} = f(p^{(N-1)})$, where $p^{(N)} \leq p_t$ and $p^{(N-1)} > p_t$, we have

$$p^{(N)} = \alpha_{N-1}\alpha_{N-2}\cdots\alpha_1\alpha_0 p_0. \tag{11}$$

Note that the value of α_k is equal to the slope of the line passing through the origin and the point $(p^{(k)}, f(p^{(k)}))$. This is illustrated in Fig. 2.

We first consider the case of determining the required number of iterations for convergence for an EXIT function that is a straight line passing through the origin, i.e., $f(p) = \alpha p$, $\alpha < 1$. In this simplest case, (11) reduces to $p^{(N)} = \alpha^N p_0$. The number of iterations required to go from p_0 to p_t can then be computed exactly:

$$N = \left\lceil \frac{\ln(p_t) - \ln(p_0)}{\ln(\alpha)} \right\rceil. \tag{12}$$

The main idea for extending the above formula to nonlinear $f(p)$ is to momentarily ignore the fact that N has to be an integer, and to compute the incremental increase in N as a function of the incremental change in p_t. The key is to recognize from (11) that the required number of iterations is a function of local slopes only. When $f(p)$ is nonlinear, the nonuniform local slopes can be used as weighting factors in an accurate estimate of N.

TABLE I
ESTIMATES FOR THE NUMBER OF ITERATIONS, $p_t = 10^{-6}$, $p_0 = 1$

$f(p)$	**Actual**	**Estimated**
$0.4p + 0.45p^2 - 1.05p^3 + 0.2p^4 + 0.2p^5 + 0.4p^6$	16	15.4
$0.7p + 0.2p^2 + 0.40p^3 - 0.4p^6$	60	59.1
$0.5p - 0.45p^2 + 0.5p^4 + 0.4p^6$	21	19.6

Fix some $w \in [p_t, p_0]$. Assume that $\alpha(p) = \frac{f(p)}{p}$ is constant for $p \in [w - \Delta w, w]$, with $\Delta w > 0$ being sufficiently small. Returning to the linear case, and considering N to be a non-integer quantity, the incremental change in N as a function of an incremental change in p_t can be obtained by taking the derivative of N with respect to p_t:

$$\frac{dN}{dp} = \frac{1}{p\ln(\alpha(p))}. \tag{13}$$

Therefore, the required number of iterations to progress from w to $w - \Delta w$ is $\Delta N(w) = \frac{-\Delta w}{w\ln(\alpha(w))}$. Thus, to estimate the total number of iterations from p_0 to p_t, we may integrate (13) and obtain

$$N \approx \int_{p_t}^{p_0} \frac{dp}{p\ln\left(\frac{p}{f(p)}\right)}. \tag{14}$$

Despite making various "continuous approximations" in its derivation, this formula provides a surprisingly accurate estimate of the required number of function evaluations, as exemplified in Table I for a set of polynomials that closely approximate the shape of typical EXIT charts. We note that while it is possible to generate functions for which the measure is arbitrarily inaccurate, for the class of EXIT-chart-like functions, such pathological cases do not arise.

Of course, to obtain $f(p)$ one must perform density evolution, from which one could directly obtain the required number of iterations to achieve a target error rate. Thus the value of the measure lies not so much in its ability to accurately predict the number of iterations but rather as an objective function for an optimization program that seeks to determine an $f(p) = \sum_i \lambda_i f_i(p)$ that minimizes N, subject to a rate constraint. This is facilitated by the fact that the above expression for N is differentiable with respect to the design parameters λ_i. More importantly, under a mild condition, (14) can be shown to be a convex function of λ_i.

Theorem 1: Let $f(p) = \sum_i \lambda_i f_i(p)$, where $f(p) < p$. The function $\int_{p_t}^{p_0} \frac{dp}{p\ln\left(\frac{p}{f(p)}\right)}$ is convex in λ_i in the region $\{\lambda_i \mid f(p) \geq e^{-2}p\}$.

Proof: Convexity of the integrand is sufficient for convexity of the integral, since the sum of convex functions is convex. Further, to show convexity of the integrand as a function of λ_i, it follows directly from the definition of convexity that one need only show convexity along all lines in λ-space that intersect its domain [21]; a function restricted to a line is univariate in $t \in \mathbb{R}$, and thus the convexity of the integrand (on its domain, $\{t \in \mathbb{R} \mid \sum_i (t\gamma_i + \psi_i) f_i(p) < p\}$) can be

verified directly by taking its second derivative. The integrand, restricted to the line, is of the form:

$$g(t) = \frac{1/p}{\Phi - \ln(t\Gamma + \Psi)} \tag{15}$$

where $\Phi = \ln(p)$, $\Gamma = \sum_i \gamma_i f_i(p)$ and $\Psi = \sum_i \psi_i f_i(p)$. Using the fact that $\Phi > \ln(t\Gamma + \Psi)$ on the domain of $g(t)$, a direct computation reveals that the second derivative with respect to t is always positive if $\Phi - \ln(t\Gamma + \Psi) \leq 2$, which is equivalent to $f(p) \geq e^{-2}p$. ∎

Note that Theorem 1 provides a sufficient condition for convexity; only when the condition is violated for all $p \in [p_t, p_0]$, does it imply that the optimization program is nonconvex.

C. Optimization Program Formulation

We are now ready to formulate the problem of minimizing the decoding complexity of a code subject to a rate constraint. The optimization variables are the variable degree distribution parameters λ_i. Implicitly, we must have $\sum_i \lambda_i = 1$ and $\lambda_i \geq 0$. It is easy to see from (1) that if the check-degree distribution ρ_i is assumed to be fixed, the code rate is simply a function of λ_i. More specifically, the rate constraint becomes a linear constraint:

$$\sum_i \lambda_i / i \geq \frac{1}{1 - R_0} \sum_j \rho_j / j. \tag{16}$$

Clearly, the above constraint would be met with equality for the minimal complexity code. In this case, the complexity measure K is then directly proportional to the number of iterations N.

The idea is now to start with some initial $\bar{\lambda}$, compute its associated EXIT functions $f_i(p)$, and update λ by solving the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \left(\frac{1 - R_0}{R_0 \sum_i \rho_i / i}\right) \int_{p_t}^{p_0} \frac{dp}{p \ln\left(\frac{p}{\sum_i \lambda_i f_i(p)}\right)} \\ \text{subject to} \quad & \sum_i \lambda_i / i \geq \frac{1}{1 - R_0} \sum_i \rho_i / i \\ & \sum_i \lambda_i = 1 \\ & \lambda_i \geq 0 \\ & ||\lambda - \bar{\lambda}||_\infty < \epsilon \end{aligned} \tag{17}$$

Here, ρ_i is fixed, R_0 is the fixed target rate, and ϵ is the maximum permissible change in any component of λ, which is set to a small number to ensure that the elementary EXIT charts $f_i(p)$ remain accurate.

In practice, the above optimization problem is solved repeatedly, with $\bar{\lambda}$ updated in each step. The initial $\bar{\lambda}$ can be set to be the variable degree distribution of the threshold-optimized code for some target channel condition. Such a distribution can be obtained by solving (8). Next, we set R_0 to be the target rate (which must be lower than the rate of the threshold-optimized code), and solve the optimization problem to obtain a minimal complexity code. However, because of the ϵ constraint, the achievable complexity reduction in one iteration is typically limited. To further reduce complexity, we set $\bar{\lambda}$ to the most recently computed λ, and repeat the optimization procedure using the updated elementary EXIT charts. This iterative process should be repeated until the complexity reduction from one iteration to the next falls below some threshold value.

D. Design Notes

While the preceding optimization program forms the core of our design procedure, several further comments are in order.

1) Convexity and Optimality: Each iteration of the design procedure consists of finding the minimum-complexity code whose degree distribution deviates from the initial condition $\bar{\lambda}$ by at most ϵ, by solving the optimization problem (17). This is done for a fixed channel condition and a given target rate R_0. Note that the constraints of (17) are linear. Thus, when the objective function of (17) is indeed convex, the optimization problem (17) belongs to a class of convex optimization problems for which the globally optimal solution can be efficiently found using standard convex optimization techniques [21]. However, from Theorem 1, we only know that convexity holds when $\{\lambda_i \mid f(p) \geq e^{-2}p\}$.

In some cases, we can argue that the sufficient condition for convexity necessarily holds, and thus by Theorem 1 the problem is convex. First, observe that for all cases of interest, when the sufficient condition for convexity condition is violated, it is violated near the origin of an EXIT chart. The convexity condition imposes a minimum slope for $f(p)$ near the origin, analogous to the usual stability condition [19], which constrains the slope near the origin to be less than 1. Under the assumption that an EXIT chart that violates the convexity condition necessarily violates the condition near the origin, one can determine the rate R_c such that an EXIT chart associated with a code of rate greater than R_c must have a slope greater than $\frac{1}{e^2}$ at $p = 0$. If the desired rate of the complexity-minimized code is greater than R_c, it is sufficient to limit the optimization to the convex region, and a globally optimal solution for (17) can be efficiently found; if the desired rate is less than R_c, then we solve the optimization problem without the $\{\lambda_i \mid f(p) \geq e^{-2}p\}$ constraint, but we are only guaranteed to find a local optimum. As a specific illustration of the preceding, consider the special case of capacity-approaching codes, where the flatness condition [22] implies[2] that the slope of the EXIT chart approaches 1 (which is clearly greater than $\frac{1}{e^2}$) as p tends to zero, therefore the problem of finding capacity-approaching complexity-minimized codes is convex; this agrees with the results of [16], which proved the convexity as the gap to capacity goes to zero.

Finally, even when the sufficient condition for convexity is not satisfied, we observed experimentally that regardless of the initial variable degree distribution, the solution of the optimization program converges to a unique distribution (for some fixed target rate, channel condition, and check degree distribution), and thus in practice it seems a globally optimal solution can always be found efficiently. Coupled with the accuracy of our measure, this suggests that solving our optimization program yields a near-minimal-complexity code in an efficient manner.

[2]Strictly speaking, this result is proved for the BEC, but a similar observation holds for more general channels

2) Setting a Target Message Error Rate: The proposed design procedure involves tracking the evolution of the average message error rate over all variable nodes. In practice, we are not directly interested in achieving a target *message*-error rate p_t, but rather a target *bit*-error rate p_b over the information bits of the code. In the following, we relate p_b with p_t and give guidelines as to how to set p_t in practice.

First, the message error rates at different variable nodes are different and they depend on the variable degrees. Further, when a decoder completes its alloted quantity of iterations, making a decision at each variable node involves combining the channel message and all d check-to-variable messages. Note that this is equivalent to the extrinsic message error rate for a degree $d+1$ variable node, which is different from that of the outgoing message of a degree d variable node, (which involves a channel message and $d-1$ check-to-variable messages.)

To compute the message error rate of variable nodes of different degrees, it is necessary to compute the node-perspective (rather than edge-perspective) variable degree distribution:

$$\Lambda(x) = \sum_{i=2}^{d_v} \Lambda_i x^i, \tag{18}$$

where

$$\Lambda_i = \frac{\frac{\lambda_i}{i}}{\sum_{j=2}^{d_v} \frac{\lambda_j}{j}}. \tag{19}$$

Furthermore, in a systematic realization of an LDPC code, one can assign information bits to high-degree variable nodes and parity-check bits to low-degree variable nodes. In the case of threshold-optimized codes, this is without loss in performance [23, Theorem 1]. Thus, we define a node-perspective variable degree distribution over the information bits as follows. Let k be such that $\sum_{i=k}^{d_v} \Lambda_i \geq R_0$ and $\sum_{i=k+1}^{d_v} \Lambda_i < R_0$, where R_0 is the target rate. The degree distribution $\Gamma(x)$ over the information bits is:

$$\Gamma(x) = \sum_{i=k}^{d_v} \Gamma_i x^i, \tag{20}$$

where

$$\Gamma_k = 1 - \frac{\sum_{j=k+1}^{d_v} \Gamma_j}{R_0}, \text{ and } \Gamma_j = \frac{\Lambda_j}{R_0} \text{ for } i > k. \tag{21}$$

Finally, for a given code and channel condition, the bit-error rate over the information bits p_b, and the average message-error rate p_t can be related as follows:

$$p_b = \mathrm{BER}(p_t) = \sum_{i=k}^{d_v} \Gamma_i f_{i+1}(p_t), \tag{22}$$

The above relation allows one to determine the appropriate p_t in code design, for a given target p_b. In each iteration of the optimization procedure, p_t can be computed based on the target p_b and the $\lambda(x)$ in the current iteration. Note that equation (22) can be easily solved numerically, as $\mathrm{BER}(p_t)$ is a strictly increasing function of p_t.

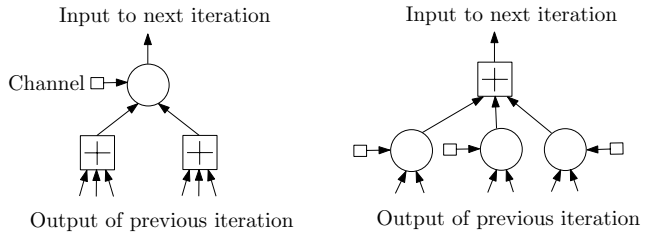


Fig. 3. Depth-one decoding trees: The (variable-perspective) EXIT charts in Section II correspond to the left-hand tree, whereas the right-hand tree corresponds to check-perspective EXIT charts.

3) Optimizing the Check-Degree Distribution: The proposed optimization program assumes a fixed check-degree distribution $\rho(x)$. Thus, finding the minimal complexity code requires an optimization over the choice of check degree distribution.

For threshold-optimized codes, there exists convincing evidence that it suffices to consider only check-degree distributions whose degrees are concentrated on two consecutive values [20], [24]. That is to say, if the average check node degree is chosen to be $n \leq \bar{d}_c < n+1$, where n is an integer, then non-zero weights exist only on degree-n and degree-$n+1$ check-nodes, and ρ_n satisfies

$$\frac{1}{\bar{d}_c} = \frac{\rho_n}{n} + \frac{1-\rho_n}{n+1}. \tag{23}$$

For complexity-minimized codes, it is possible to use a procedure similar to the proposed complexity-optimization program to provide further evidence in favor of concentrated check distributions. The idea is to fix a target rate R_0 and to fix $\lambda(x)$, and to show that the complexity-minimizing $\rho(x)$ has a concentrated distribution. Note that fixing R_0 and $\lambda(x)$, the average check-degree is fixed, therefore minimizing the complexity is equivalent to minimizing the number of decoding iterations. It is possible to use an approach analogous to that outlined in Section II-B to define a check-perspective EXIT chart (i.e., one that tracks the message error rate of check-to-variable messages, see Fig. 3). One can then express the decoding complexity as a function of $\rho(x)$ and the elementary check-perspective EXIT charts, and a suitable optimization program can be formulated to find the complexity-minimizing check-degree distribution. In all investigated cases, we found that the resulting check-degree distribution was indeed concentrated on two consecutive degrees.

Finally, in many cases it suffices to consider only regular check degrees, i.e. $\rho_n = 1$ for some n, with negligible performance degradation. Note that with regular check degrees, the capacity of the BEC can be achieved [22], and for the Gaussian channel, performance very close to the Shannon limit has also been reported [5].

IV. Code Design and Simulation Results

Given a target rate and a target decoding complexity, the best code is the one which permits successful decoding (with the target complexity) on the worst channel. In other words, it is the highest threshold code that satisfies the design specifications. In this section, we illustrate how to

find such codes using the optimization program presented in Section III, and compare their performance, via simulation, to traditional threshold-optimized codes, which were designed using `LdpcOpt` [25]. A binary-input AWGN channel with noise variance σ^2 and a target $p_b = 10^{-6}$ are assumed, and the sum-product algorithm is used at the decoder.

A. Design Example

To illustrate our optimization method, we design codes of rate one-half for different target decoding complexities, with a maximum variable node degree of 30.[3]

First, from the given specifications, the threshold-optimized code can be found by well-known methods [1]–[3]. Using `LdpcOpt`, the threshold-optimized code $\mathcal{C}_T$ has decoding threshold $\sigma = 0.9713$, $\rho(x) = x^8$, and

$$\begin{aligned}\lambda(x) &= 0.21236x + 0.19853x^2 + 0.00838x^4 \\ &\quad +0.07469x^5 + 0.01424x^6 + 0.16652x^7 \\ &\quad +0.00912x^8 + 0.02002x^9 + 0.00025x^{19} \\ &\quad +0.29589x^{29}.\end{aligned}$$

Note that finding the threshold-optimized code implicitly involves a search over $\rho(x)$.

For $\rho(x) = x^8$ and any $\sigma < 0.9713$, we can solve our optimization program (17) to find the minimal complexity code of rate one-half; that is, the optimization yields a unique complexity-minimized code for each channel condition. In essence, we obtain a curve of the optimized complexity vs. the channel condition. Now, as discussed in Section III, minimizing the decoding complexity at a fixed channel condition is equivalent to minimizing $N\bar{d}_c$, and therefore involves a search over the average check degree. Thus, in a manner analogous to the case of $\rho(x) = x^8$, we compute the K vs. σ curve for various $\bar{d}_c$ (from an appropriately chosen set, centered about $d_c = 9$), and plot the curves on a single graph. The resulting graph is illustrated in Fig. 4, where check distributions of the form $\rho(x) = x^{\bar{d}_c - 1}$ are considered. Note that a slightly more refined result is possible if one considers check distributions concentrated on consecutive degrees.

Now, for a target decoding complexity, it is straightforward to use Fig. 4 to determine the parameters of the best code. This is accomplished by finding the largest σ at which one of the curves achieves the target complexity, and finding the complexity-minimized code for the corresponding check degree and noise variance. We present below the resulting codes for three target complexities.

For $K = 150$, we have $d_c = 6$ and $\sigma = 0.855$, and the resulting code $\mathcal{C}_{150}$ has

$$\lambda(x) = 0.02799x + 0.94752x^2 + 0.02449x^6.$$

For $K = 400$, we have $d_c = 8$ and $\sigma = 0.913$, and the resulting code $\mathcal{C}_{400}$ has

$$\begin{aligned}\lambda(x) &= 0.10733x + 0.50459x^2 + 0.07627x^{12} \\ &\quad +0.31181x^{13}.\end{aligned}$$

[3]This is sufficient such that threshold-optimized codes approach the BI-AWGN capacity to within 0.07dB.

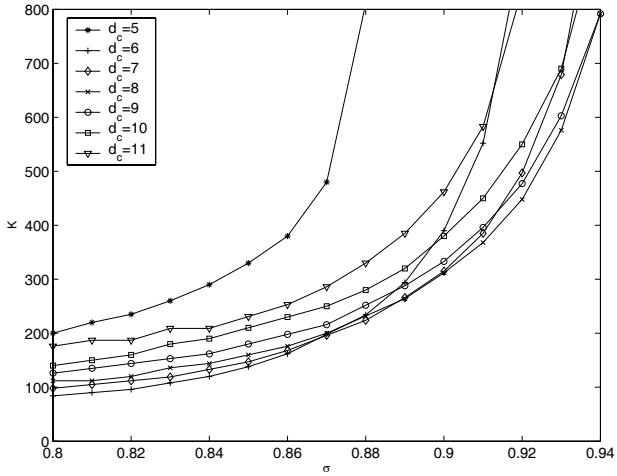


Fig. 4. Minimum decoding complexity as a function of channel condition, for various check degrees, with $R = 0.5$ and maximum variable degree 30.

For $K = 800$, we have $d_c = 8$ and $\sigma = 0.939$, and the resulting code $\mathcal{C}_{800}$ has

$$\begin{aligned}\lambda(x) &= 0.16627x + 0.40111x^2 + 0.08803x^{10} + 0.17446x^{11} \\ &\quad +0.17013x^{15}.\end{aligned}$$

B. Simulation Results

We compare the performance of the complexity-minimized codes with that of threshold-optimized codes. For each degree distribution, we design a parity-check matrix of length $n = 10^5$. We ensure that the corresponding graph does not have any cycles of length 4, but the matrix is otherwise randomly constructed. We note that for short-block-lengths, graph-structure optimization via progressive-edge-growth [26], and its variants (see [27] and references therein), significantly improve performance. However, for long-block-lengths, these algorithms are computationally intensive, and their benefits are reduced.

The complexity-minimized codes are evaluated for their target decoding complexities, while the performance of the threshold-optimized codes are presented for a range of decoding complexities. The bit-error-rates (BER), computed over the information bits of the code, as a function of signal-to-noise (SNR) ratio, are plotted.

While it is traditional to discuss the coding gain (in dB) of one coding scheme relative to another, we prefer to evaluate the complexity-reduction afforded by our complexity-minimized codes. Fig. 5 shows the BER curves for $\mathcal{C}_{150}$, $\mathcal{C}_{400}$ and $\mathcal{C}_{800}$, as well as $\mathcal{C}_T$. Each of the complexity-optimized codes is evaluated for its target decoding complexity, while the performance of the threshold-optimized code is presented for a range of decoding iterations. For N decoding iterations, the decoding complexity of the threshold-optimized code is $K = 9N$.

For a BER $p_b = 10^{-6}$, $\mathcal{C}_T$ requires N to be slightly greater than 30, which translates to a decoding complexity $K = 270$, whereas $\mathcal{C}_{150}$ achieves the target with $K = 150$; thus, the complexity-optimized code provides a 44% reduction in decoding complexity at the same SNR.

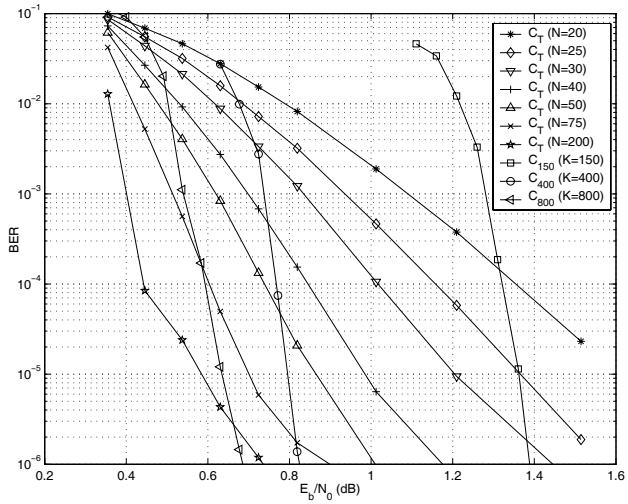


Fig. 5. Performance of $\mathcal{C}_{150}$, $\mathcal{C}_{400}$, $\mathcal{C}_{800}$, each evaluated for its target decoding complexity, and $\mathcal{C}_T$ for various numbers of decoding iterations. $R = 0.5$. Blocklength $n = 10^5$. Note that for $\mathcal{C}_T$ with N decoding iterations, $K = 9N$.

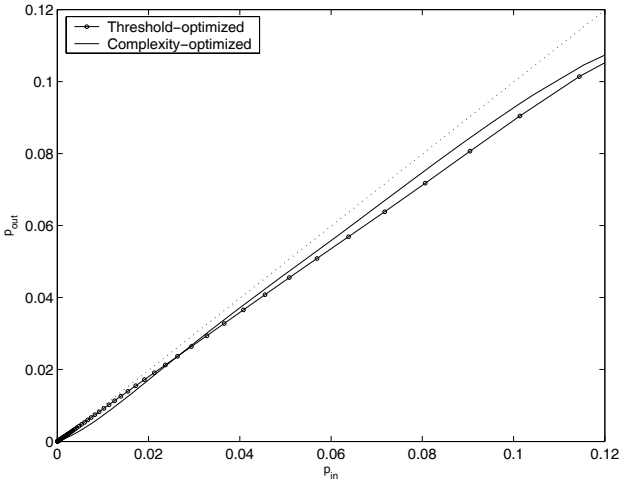


Fig. 6. EXIT charts for $\mathcal{C}_{150}$ and $\mathcal{C}_T$ on AWGN channel, $\sigma = 0.9$, in linear scale.

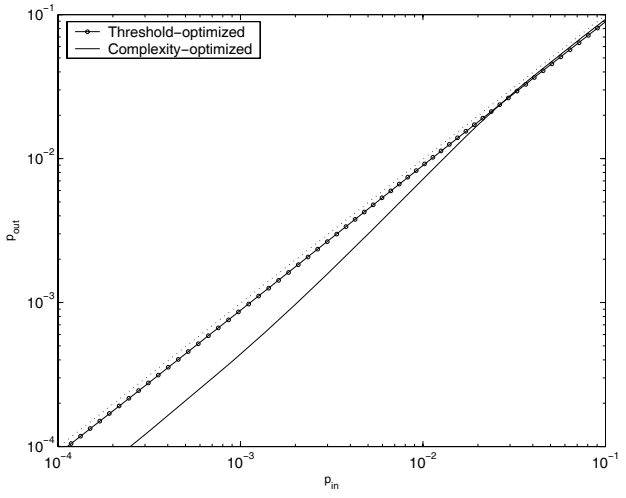


Fig. 7. EXIT charts for $\mathcal{C}_{150}$ and $\mathcal{C}_T$ on AWGN channel, $\sigma = 0.9$, in log scale.

Of course, as we increase the decoding complexity, the SNR to achieve the target BER will decrease. Relative to $\mathcal{C}_{400}$, which achieves $p_b = 10^{-6}$ at $E_b/N_0 = 0.84$dB, $\mathcal{C}_T$ requires N to be slightly greater than 75, which translates to a decoding complexity $K = 675$; in this case, the complexity-optimized code provides a 41% reduction in decoding complexity at the same SNR.

Relative to $\mathcal{C}_{800}$, which achieves $p_b = 10^{-6}$ at $E_b/N_0 = 0.69$dB, $\mathcal{C}_T$ requires N to be greater than 200, which translates to a decoding complexity $K = 1800$; in this case, the complexity-optimized code provides more than a 56% reduction in decoding complexity at the same SNR.

Finally, since the check degree distribution of a complexity-minimized code is a function of its target decoding complexity, we investigated the performance of threshold-optimized codes with reduced check degrees, and found that reducing the check degree of threshold-optimized codes (from $d_c = 9$, which maximizes the threshold) does not improve their performance-complexity tradeoff. This justifies the comparison of our codes to $\mathcal{C}_T$.

C. EXIT Chart Interpretation of the Results

To further explain the observed results, Fig. 6 shows the EXIT charts of the threshold-optimized code and the $K = 150$ complexity-optimized code, for $\sigma = 0.9$. The behavior of the EXIT charts near the origin on a log-scale is plotted in Fig. 7. For a target message error rate $p_t = 10^{-4}$, $\mathcal{C}_{150}$ requires 35 decoding iterations ($K = 280$), while $\mathcal{C}_T$ requires 63 decoding iterations ($K = 567$); this translates to a reduction in decoding complexity of approximately 50%.

An important observation is that the complexity-optimized code has a more closed EXIT chart in the earlier iterations and a more open EXIT chart at the later iterations. It is shown in [28] that, for a wide class of decoding algorithms, when the EXIT chart of two codes $f_{\mathcal{C}_A}(p)$ and $f_{\mathcal{C}_B}(p)$ satisfy $f_{\mathcal{C}_A}(p) \leq f_{\mathcal{C}_B}(p)$, $\forall p \in (0, p_0]$, then $\mathcal{C}_B$ has a higher rate than $\mathcal{C}_A$ (for the BEC, this also follows from [29]). In this example, since both codes have the same rate, one EXIT chart cannot dominate the other one everywhere, but the optimization program carefully trades the area underneath the EXIT chart for the complexity. Furthermore, for the complexity-minimized codes, the delayed appearance of the waterfall region is due to the fact that their EXIT charts are initially less open than the EXIT chart of the threshold-optimized code. On the other hand, the openness of the complexity-minimized EXIT charts near the origin, translates to the steeper waterfalls in the above figures. These effects can be attributed to the decreased fraction of degree-two nodes in the complexity-minimized codes. It is well known that the minimum distance [30] and the size of the smallest stopping set [31] are strongly dependent on the fraction of degree-two nodes; therefore, it is not surprising that the complexity-minimized codes do not appear to suffer from error floors, even when the number of decoding iterations is small.

D. Discussion

The effectiveness of our design methodology stems from the accuracy of our measure of complexity, but it also rests upon the accuracy of density evolution analysis. At short-block-lengths, where the performance of irregular LDPC codes are sensitive to the structure of the parity-check matrix, and the underlying assumptions for density evolution are not valid, our complexity-optimized codes provide very similar performance to threshold-optimized codes. We note that it may be possible to further improve upon our results (both at short and long block lengths) by optimizing over a more general ensemble of LDPC codes. Specifically, multi-edge type LDPC codes [32] generalize the irregular ensemble to include various edge types, thus providing greater control over graph structure; most importantly, this structure translates to short block length performance that more closely approaches the asymptotic predictions of density evolution. Similarly, ARA codes generally provide improved short block length performance, relative to an LDPC code of the same threshold. However, in order to apply our complexity-minimization method to either family of codes, an EXIT chart-based design procedure would be required; to our knowledge, no EXIT chart-based design procedure has been developed for either family of codes.

V. Concluding Remarks

This paper presents a methodology for the design of irregular LDPC codes with an optimized complexity-rate tradeoff. Our methodology is based on a new measure of complexity, which accurately models the required number of iterations for convergence in the iterative decoding process, and a novel formulation of a complexity-minimization problem. A sufficient condition for convexity of the complexity-optimization problem in the variable edge-degree distribution is presented; when it is not satisfied, numerical experiments nevertheless suggest that the local minimum is unique. The effectiveness of our design methodology stems from the accuracy of our measure of complexity, but also rests upon the accuracy of density evolution analysis. When the block lengths of the codes are long enough such that density evolution provides an accurate prediction of decoding trajectory, our complexity-minimized codes provide a superior performance-complexity tradeoff as compared to threshold-optimized codes, within the ensemble of irregular codes.

Acknowledgment

We acknowledge the anonymous reviewers for their detailed comments, which improved the presentation of the paper.

References

[1] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585-598, Feb. 2001.

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

[3] S.-Y. Chung, G. D. Forney Jr., T. J. Richardson, and R. L. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58-60, Feb. 2001.

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

[5] M. Ardakani and F. R. Kschischang, "A more accurate one-dimensional analysis and design of LDPC codes," *IEEE Trans. Commun.*, vol. 52, no. 12, pp. 2106-2114, Dec. 2004.

[6] 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.

[7] 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.

[8] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.

[9] D. J. Costello and G. D. Forney Jr., "Channel coding: the road to channel capacity," *Proc. IEEE*, vol. 95, no. 6, pp. 1150-1177, June 2007.

[10] A. Khandekar and R. J. McEliece, "On the complexity of reliable communication on the erasure channel," in *Proc. IEEE International Symp. Inf. Theory*, 2001, p. 1.

[11] I. Sason and G. Wiechman, "Bounds on the number of iterations for turbo-like ensembles over the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2602-2617, June 2009.

[12] I. Sason and R. Urbanke, "Parity-check density versus performance of binary linear block codes over memoryless symmetric channels," *IEEE Trans. Inf. Theory*, vol. 49, no. 7, pp. 1611-1635, July 2003.

[13] ——, "Complexity versus performance of capacity-achieving irregular repeat-accumulate codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 50, no. 6, pp. 1247-1256, June 2004.

[14] H. Pfister, I. Sason, and R. Urbanke, "Capacity-achieving ensembles for the binary erasure channel with bounded complexity," *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp. 2352-2379, July 2005.

[15] H. D. Pfister and I. Sason, "Accumulate-repeat-accumulate codes: capacity-achieving ensembles of systematic codes for the erasure channel with bounded complexity," *IEEE Trans. Inf. Theory*, vol. 53, no. 6, pp. 2088-2115, June 2007.

[16] X. Ma and E.-H. Yang, "Low-density parity-check codes with fast decoding convergence speed," in *Proc. IEEE International Symp. Inf. Theory*, Chicago, USA, June 2004, p. 103.

[17] W. Yu, M. Ardakani, B. Smith, and F. R. Kschischang, "Complexity-optimized LDPC codes for Gallager decoding algorithm B," in *Proc. IEEE International Symp. Inf. Theory*, Adelaide, Australia, Sept. 2005.

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

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

[20] S.-Y. Chung, T. J. Richardson, and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 657-670, Feb. 2001.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[22] A. Shokrollahi, "New sequences of linear time erasure codes approaching the channel capacity," in *Proc. International Symp. Applied Algebra, Algebraic Algorithms Error-Correcting Codes*, no. 1719, Lecture Notes in Computer Science, 1999, pp. 65-67.

[23] T. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 638–656, Feb. 2001.

[24] L. Bazzi, T. J. Richardson, and R. L. Urbanke, "Exact thresholds and optimal codes for the binary-symmetric channel and Gallager's decoding algorithm A," *IEEE Trans. Inf. Theory*, vol. 50, no. 9, pp. 2010-2021, Sept. 2004.

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

[26] E.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," *IEEE Trans. Inf. Theory*, vol. 51, pp. 386-398, Jan. 2005.

[27] E. Sharon and S. Litsyn, "Constructing LDPC codes by error minimization progressive edge growth," *IEEE Trans. Commun.*, vol. 56, pp. 359-368, Mar. 2008.

[28] M. Ardakani, T. H. Chan, and F. R. Kschischang, "EXIT-Chart properties of the highest-rate LDPC code with desired convergence behavior," *IEEE Commun. Lett.*, vol. 9, no. 1, pp. 52-54, Jan. 2005.

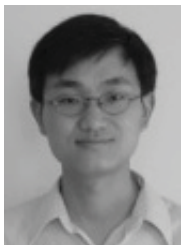
[29] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: model and erasure channel properties," *IEEE Trans. Inf. Theory*, vol. 50, pp. 2657-2673, Nov. 2004.

[30] C. Di, T. J. Richardson, and R. L. Urbanke, "Weight distribution of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 11, pp. 4839-4855, Nov. 2006.
[31] A. Orlitsky, K. Viswanathan, and J. Zhang, "Stopping set distribution of LDPC code ensembles," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, pp. 929-953, Mar. 2005.
[32] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge, UK: Cambridge University Press, 2008.

Benjamin Smith received the B.A.Sc. degree from the University of Windsor, Windsor, ON, Canada, in 2004, and the M.A.Sc. degree from the University of Toronto, Toronto, ON, Canada, in 2006, both in electrical engineering. He is currently working toward the Ph.D. degree in electrical engineering at the University of Toronto. His current research interests include coding and information theory for fiber-optic communication systems. He is a recipient of the Natural Sciences and Engineering Research Council (NSERC) of Canada Postgraduate Scholarship.

Frank R. Kschischang received the B.A.Sc. degree (with honors) from the University of British Columbia, Vancouver, BC, Canada, in 1985 and the M.A.Sc. and Ph.D. degrees from the University of Toronto, Toronto, ON, Canada, in 1988 and 1991, respectively, all in electrical engineering. He is a Professor of Electrical and Computer Engineering and Canada Research Chair in Communication Algorithms at the University of Toronto, where he has been a faculty member since 1991. During 1997-98, he was a visiting scientist at MIT, Cambridge, MA and in 2005 he was a visiting professor at the ETH, Zurich. His research interests are focused on the area of channel coding techniques.

He was the recipient of the Ontario Premier's Research Excellence Award. From 1997-2000, he served as an Associate Editor for Coding Theory for the IEEE TRANSACTIONS ON INFORMATION THEORY. He also served as technical program co-chair for the 2004 IEEE International Symposium on Information Theory (ISIT), Chicago, and as general co-chair for ISIT 2008, Toronto.



Wei Yu (S'97-M'02) received the B.A.Sc. degree in Computer Engineering and Mathematics from the University of Waterloo, Waterloo, Ontario, Canada in 1997 and M.S. and Ph.D. degrees in Electrical Engineering from Stanford University, Stanford, CA, in 1998 and 2002, respectively. Since 2002, he has been with the Electrical and Computer Engineering Department at the University of Toronto, Toronto, Ontario, Canada, where he is now an Associate Professor and holds a Canada Research Chair in Information Theory and Digital Communications. His main research interests include multiuser information theory, optimization, wireless communications and broadband access networks.

Prof. Wei Yu is currently an Editor for IEEE TRANSACTIONS ON COMMUNICATIONS. He was an Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2004 to 2007, a Guest Editor of IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS for a special issue on "Nonlinear Optimization of Communications Systems" in 2006, and a Guest Editor of EURASIP JOURNAL ON APPLIED SIGNAL PROCESSING for a special issue on "Advanced Signal Processing for Digital Subscriber Lines" in 2005. He is a member of the Signal Processing for Communications Technical Committee of the IEEE Signal Processing Society. He received the IEEE Signal Processing Society 2008 Best Paper Award, the McCharles Prize for Early Career Research Distinction in 2008, the Early Career Teaching Award from the Faculty of Applied Science and Engineering, University of Toronto in 2007, and the Early Researcher Award from Ontario in 2006. He is a registered Professional Engineer in Ontario.

Masoud Ardakani received the B.Sc. degree from Isfahan University of Technology in 1994, the M.Sc. degree from Tehran University in 1997 and the Ph.D. degree from the University of Toronto in 2004, all in Electrical Engineering. He was a Postdoctoral fellow at the University of Toronto from 2004 to 2005. He is currently an Associate Professor of Electrical and Computer Engineering and Alberta Ingenuity New Faculty at the University of Alberta, where he holds an Informatics Circle of Research Excellence (iCORE) Junior Research Chair in wireless communications. His research interests are in the general area of digital communications, codes defined on graphs and iterative decoding techniques. He serves as an Associate Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and the IEEE COMMUNICATION LETTERS.