Design Methods for Irregular Repeat–Accumulate Codes

Aline Roumy, *Member, IEEE*, Souad Guemghar, *Student Member, IEEE*, Giuseppe Caire, *Senior Member, IEEE*, and Sergio Verdú, *Fellow, IEEE*

***Abstract*—We optimize the random-like ensemble of irregular repeat–accumulate (IRA) codes for binary-input symmetric channels in the large block-length limit. Our optimization technique is based on approximating the evolution of the densities (DE) of the messages exchanged by the belief-propagation (BP) message-passing decoder by a one-dimensional dynamical system. In this way, the code ensemble optimization can be solved by linear programming. We propose four such DE approximation methods, and compare the performance of the obtained code ensembles over the binary-symmetric channel (BSC) and the binary-antipodal input additive white Gaussian noise channel (BIAWGNC). Our results clearly identify the best among the proposed methods and show that the IRA codes obtained by these methods are competitive with respect to the best known irregular low-density parity-check (LDPC) codes. In view of this and the very simple encoding structure of IRA codes, they emerge as attractive design choices.**

***Index Terms*—Belief propagation (BP), channel capacity, density evolution, low-density parity-check (LDPC) codes, stability, threshold, turbo codes.**

I. Introduction

SINCE the discovery of turbo codes [1], there have been several notable inventions in the field of random-like codes. In particular, the rediscovery of the low-density parity-check (LDPC) codes, originally proposed in [2], the introduction of irregular LDPCs [3], [4], and the introduction of the repeat-accumulate (RA) codes [5].

In [3], [4], irregular LDPCs were shown to asymptotically achieve the capacity of the binary erasure channel (BEC) under iterative message-passing decoding. Although the BEC is the only channel for which such a result currently exists, irregular LDPC codes have been designed for other binary-input channels (e.g., the binary-symmetric channel (BSC), the binary-antipodal input additive white Gaussian noise channel (BIAWGNC) [6], and the binary-input intersymbol interference (ISI) channel [7]–[9]) and have shown to achieve very good performance.

First attempts to optimize irregular LDPC codes ([10] for the BEC and other channels [11]) with the density evolution (DE) technique computes the expected performance for a random-like code ensemble in the limit of infinite code block length. In order to reduce the computational burden of ensemble optimization based on the DE, faster techniques have been proposed, based on the approximation of the DE by a one-dimensional dynamical system (recursion). These techniques are exact only for the BEC (for which DE is one-dimensional). The most popular techniques proposed so far are based on the Gaussian approximation (GA) of messages exchanged in the message-passing decoder. GA in addition to the symmetry condition of message densities implies that the Gaussian density of messages is expressed by a single parameter. Techniques differ in the parameter to be tracked and in the mapping functions defining the dynamical system [12]–[18].

The introduction of irregular LDPCs motivated other schemes such as irregular RA (IRA) [19], for which similar results exist (achievability of the BEC capacity) and irregular turbo codes [20]. IRA codes are, in fact, special subclasses of both irregular LDPCs and irregular turbo codes. In IRA codes, a fraction f_i of information bits is repeated i times, for $i = 2, 3, \ldots$. The distribution

$$\left\{ f_i \geq 0, i = 2, 3, \ldots : \sum_{i=2}^{\infty} f_i = 1 \right\}$$

is referred to as the *repetition profile*, and it is kept as a degree of freedom in the optimization of the IRA ensemble. After the repetition stage, the resulting sequence is interleaved and input to a recursive finite-state machine (called accumulator) which outputs one bit for every a input symbols, where a is referred to as *grouping factor* and is also a design parameter.

IRA codes are an appealing choice because the encoder is extremely simple, their performance is quite competitive with that of turbo codes and LDPCs, and they can be decoded with a very-low-complexity iterative decoding scheme.

The only other work that has proposed a method to design IRA codes is [19], [21] where the design focuses on the choice of the grouping factor and the repetition profile. The recursive finite-state machine is the simplest one which gives full freedom to choose any rational number between 0 and 1 as the coding rate. We will also restrict our study to IRAs that use the same simple recursion of [19], although it might be expected that better codes can be obtained by including the finite-state machine as a degree of freedom in the overall ensemble optimization. The method used in [19] to choose the repetition profile was based on the infinite-block-length GA of message-passing decoding proposed in [14]. In this work, we propose and compare four low-complexity ensemble

Manuscript received October 22, 2002; revised April 1, 2004.

A. Roumy is with IRISA-INRIA, 35042 Rennes, France (e-mail: aline.roumy@irisa.fr).

S. Guemghar and G. Caire are with the Eurecom Institute, 06904 Sophia-Antipolis, France (e-mail: Souad.Guemghar@eurecom.fr; Giuseppe.Caire@eurecom.fr).

S. Verdú is with the Department of Electrical Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: verdu@princeton.edu).

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

Digital Object Identifier 10.1109/TIT.2004.831778

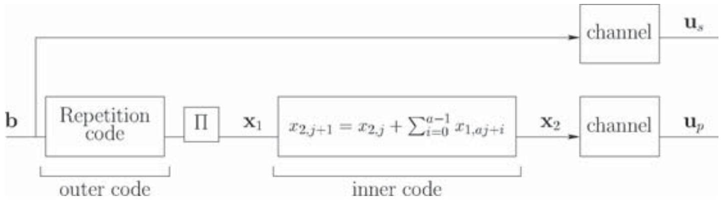

Fig. 1. IRA encoder.

optimization methods. Our approach to design IRAs is based on several tools that have been noticed recently: the EXtrinsic mutual Information Transfer (EXIT) function and its analytical properties [12], [22], [23], reciprocal channel (duality) approximation [22], [24], and the nonstrict convexity of mutual information.

The rest of the paper is organized as follows. Section II presents the systematic IRA encoder and its related decoder: the belief-propagation (BP) message-passing algorithm. Existing results on the analysis of the decoder (i.e., DE technique) are summarized and applied to the IRA code ensemble. This leads to a two-dimensional dynamical system whose state is defined on the space of symmetric distributions, for which we derive a local stability condition. In Section III, we propose a general framework in order to approximate the DE (defined on the space of distributions) by a standard dynamical system defined on the reals. We propose four low-complexity ensemble optimization methods as special cases of our general framework. These methods differ by the way the message densities and the BP transformations are approximated:

1) GA, with reciprocal channel (duality) approximation;
2) BEC approximation, with reciprocal channel approximation;
3) GA, with EXIT function of the inner decoder;
4) BEC approximation, with EXIT function of the inner decoder.

All four methods lead to optimization problems solvable by linear programming. In Section IV, we show that the first proposed method yields a one-dimensional DE approximation with the same stability condition as the exact DE, whereas the exact stability condition must be added to the ensemble optimization as an explicit additional constraint for the second method. Then, we show that, in general, the GA methods are optimistic, in the sense that there is no guarantee that the optimized rate is below capacity. On the contrary, we show that for the BEC approximation methods rates below capacity are guaranteed. In Section V, we compare our code optimization methods by evaluating their iterative decoding threshold (evaluated by the exact DE) over the BIAWGNC and the BSC.

II. Encoding, Decoding, and Density Evolution

Fig. 1 shows the block diagram of a systematic IRA encoder. A block of information bits $\boldsymbol{b} = (b_1, \ldots, b_k) \in \mathbb{F}_2^k$ is encoded by an (irregular) repetition code of rate k/n. Each bit b_j is repeated r_j times, where $(r_1, \ldots, r_k)$ is a sequence of integers

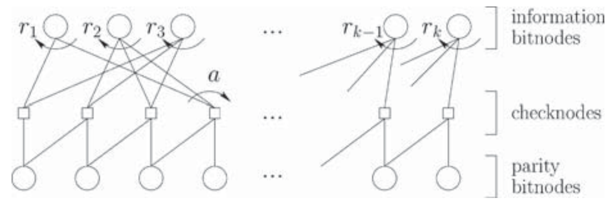

Fig. 2. Tanner graph of an IRA code.

and the resulting block $\boldsymbol{x}_1 = (x_{1,1}, \ldots, x_{1,n}) \in \mathbb{F}_2^n$ is encoded by an *accumulator*, defined by the recursion

$$x_{2,j+1} = x_{2,j} + \sum_{i=0}^{a-1} x_{1,aj+i}, \qquad j = 0, \ldots, m-1 \tag{1}$$

with initial condition $x_{2,0} = 0$, where $\boldsymbol{x}_2 = (x_{2,1}, \ldots, x_{2,m}) \in \mathbb{F}_2^m$ is the accumulator output block corresponding to the input $\boldsymbol{x}_1$, $a \geq 1$ is a given integer (referred to as *grouping factor*), and we assume that $m = n/a$ is an integer. Finally, the codeword corresponding to the information block $\boldsymbol{b}$ is given by $\boldsymbol{x} = (\boldsymbol{b}, \boldsymbol{x}_2)$.

The transmission channel is memoryless, binary-input, and symmetric-output, i.e., its transition probability $p_{Y|X}(y|x)$ satisfies

$$p_{Y|X}(y|0) = p_{Y|X}(-y|1) \tag{2}$$

where $y \mapsto -y$ indicates a *reflection* of the output alphabet.[1]

IRA codes are best represented by their Tanner graph [25] (see Fig. 2). In general, the Tanner graph of a linear code is a bipartite graph whose node set is partitioned into two subsets: the *bitnodes*, corresponding to the coded symbols, and the *checknodes*, corresponding to the parity-check equations that codewords must satisfy. The graph has an edge between bitnode α and checknode β if the symbol corresponding to α participates in the parity-check equation corresponding to β.

Since the IRA encoder is systematic (see Fig. 1), it is useful to further classify the bitnodes into two subclasses: the information bitnodes, corresponding to information bits, and the parity bitnodes, corresponding to the symbols output by the accumulator. Those information bits that are repeated i times are represented by bitnodes with degree i, as they participate in i parity-check equations. Each checknode is connected to a information bit nodes and to two parity bitnodes and represents one of the equations (for a particular j) (1). The connections between checknodes and information bitnodes are determined by the interleaver and are highly randomized. On the contrary, the connections between checknodes and parity bitnodes are arranged in a regular zig-zag pattern since, according to (1), every pair of consecutive parity bits are involved in one parity-check equation.

A random IRA code ensemble with parameters $(\{\lambda_i\}, a)$ and (information) block length k is formed by all graphs of the form of Fig. 2 with k information bitnodes, grouping factor a, and $\lambda_i n$ edges connected to information bitnodes of degree i, for $i = 2, \ldots, d$. The sequence of nonnegative coefficients $\{\lambda_i\}$ such that $\sum_{i=2}^{d} \lambda_i = 1$ is referred to as the *degree distribution* of the ensemble. The probability distribution over the code ensemble is induced by the uniform probability over all interleavers (permutations) of n elements.

The information bitnodes average degree is given by $\bar{d} \triangleq 1/(\sum_{i=2}^{d} \lambda_i/i)$. The number of edges connecting information bitnodes to checknodes is $n = k/(\sum_{i=2}^{d} \lambda_i/i)$. The number of parity bitnodes is $m = k/(a\sum_{i=2}^{d} \lambda_i/i)$. Finally, the code rate is given by

$$R = \frac{k}{k+m} = \frac{a\sum_{i=2}^{d} \lambda_i/i}{1 + a\sum_{i=2}^{d} \lambda_i/i} = \frac{a}{a+\bar{d}}. \tag{3}$$

Under the constraints $0 \le \lambda_i \le 1$ and $\sum_{i \ge 2} \lambda_i = 1$, we get $\bar{d} \ge 2$. Therefore, the highest rate with parameter a set to 1 is $1/3$. This motivates the use of $a \ge 2$ in order to get higher rates.

A. Belief Propagation Decoding of IRA Codes

In this work, we consider BP message-passing decoding [26]–[28]. In message-passing decoding algorithms, the graph nodes receive messages from their neighbors, compute new messages, and forward them to their neighbors. The algorithm is defined by the code Tanner graph, by the set on which messages take on values, by the node computation rules, and by the node activation scheduling.

In BP decoding, messages take on values in the extended real line $\mathbb{R} \cup \{-\infty, \infty\}$. The BP decoder is initialized by setting all messages output by the checknodes equal to zero. Each bitnode α is associated with the *channel observation* message (log-likelihood ratio)

$$u_\alpha = \log \frac{p_{Y|X}(y_\alpha | x_\alpha = 0)}{p_{Y|X}(y_\alpha | x_\alpha = 1)} \tag{4}$$

where y_α is the channel output corresponding to the transmission of the code symbol x_α.

The BP node computation rules are given as follows. For a given node, we identify an adjacent edge as *outgoing* and all other adjacent edges as *incoming*. Consider a bitnode α of degree i and let $m_1, \ldots, m_{i-1}$ denote the messages received from the $i-1$ incoming edges and u_α the associated channel observation message. The message $m_{o,\alpha}$ passed along the outgoing edge is given by

$$m_{o,\alpha} = m_1 + \cdots + m_{i-1} + u_\alpha. \tag{5}$$

Consider a checknode β of degree i and let $m_1, \ldots, m_{i-1}$ denote the messages received from the $i-1$ incoming edges. The message $m_{o,\beta}$ passed along the outgoing edge is given by

where the mapping $\gamma: \mathbb{R} \to \mathbb{F}_2 \times \mathbb{R}_+$ is defined by [11]

$$\gamma(z) = \left(\operatorname{sign}(z), -\log\tanh\frac{|z|}{2}\right) \tag{7}$$

and where the sign function is defined as [11]

$$\operatorname{sign}(z) = \begin{cases} 0, & \text{if } z > 0 \\ 0, & \text{with probability } 1/2 \text{ if } z = 0 \\ 1, & \text{with probability } 1/2 \text{ if } z = 0 \\ 1, & \text{if } z < 0. \end{cases}$$

Since the code Tanner graph has cycles, different schedulings yield in general nonequivalent BP algorithms. In this work, we shall consider the following "classical" schedulings.

- LDPC-like scheduling [19]. In this case, all bitnodes and all checknodes are activated alternately and in parallel. Every time a node is activated, it sends outgoing messages to all its neighbors. A decoding iteration (or "round" [31]) consists of the activation of all bitnodes and all checknodes.
- Turbo-like scheduling. Following [29], a good decoding scheduling consists of isolating large trellis-like subgraphs (or, more generally, normal realizations in Forney's terminology) and applying locally the forward–backward Bahl–Cocke–Jelinek–Raviv (BCJR) algorithm [30] (that implements efficiently the BP algorithm on normal cycle-free graphs), as done for turbo codes [1]. A decoding iteration consists of activating all the information bitnodes in parallel (according to (5)) and of running the BCJR algorithm over the entire accumulator trellis. In particular, the checknodes do not send messages to the information bitnodes until the BCJR iteration is completed.

Notice that for both of the above schedulings one decoder iteration corresponds to the activation of all information bitnodes in the graph exactly once.

B. Density Evolution and Stability

The bit-error rate (BER) performance of BP decoding averaged over the IRA code ensemble and over the noise observations can be analyzed, for any finite number ℓ of iterations and in the limit of $k \to \infty$, by the DE technique [11]. The usefulness of the DE method stems from the *Concentration Theorem* [31], [10] which guarantees that, with high probability, the BER after ℓ iterations of the BP decoder applied to a randomly selected code in the ensemble and to a randomly generated channel noise sequence is close to the BER computed by DE, for sufficiently large block length.

Next, we formulate the DE for IRA codes and we study the stability condition of the fixed-point corresponding to zero BER. As in [11, Sec. III-B], we introduce the space of *distributions* whose elements are nonnegative nondecreasing

It can be shown that, for a binary-input symmetric-output channel, the distributions of messages at any iteration of the DE satisfy the symmetry condition

$$\int h(x)dF(x) = \int e^{-x}h(-x)dF(x) \tag{8}$$

for any function h for which the integral exists. If F has density f, (8) is equivalent to

$$f(x) = e^{x}f(-x). \tag{9}$$

With some abuse of terminology, distributions satisfying (8) are said to be *symmetric*. The space of symmetric distributions will be denoted by $\mathcal{F}_{\rm sym}$.

The BER operator Pe: $\mathcal{F}_{\rm sym} \to [0, 1/2]$ is defined by

$$\mathrm{Pe}(F) = \frac{1}{2}(F^{-}(0) + F(0))$$

where $F^{-}(z)$ is the left-continuous version of $F(z)$. We introduce the "delta at zero" distribution, denoted by Δ_0, for which $\mathrm{Pe}(\Delta_0) = 1/2$, and the "delta at infinity" distribution, denoted by Δ_∞, for which $\mathrm{Pe}(\Delta_\infty) = 0$.

The symmetry property (8) implies that a sequence of symmetric distributions $\{F^{(\ell)}\}_{\ell=0}^{\infty}$ converges to Δ_∞ if and only if $\lim_{\ell\to\infty} \mathrm{Pe}(F^{(\ell)}) = 0$, where convergence of distributions is in the sense given in [11, Sec. III-F].

The DE for IRA code ensembles is given by the following proposition whose derivation is omitted as it is completely analogous to the derivation of DE in [11] for irregular LDPC codes.

Proposition 1: Let P_ℓ (respectively, $\widetilde{P}_\ell$) denote the average distribution of messages passed from an information bitnode (respectively, parity bitnode) to a checknode, at iteration ℓ. Let Q_ℓ (respectively, $\widetilde{Q}_\ell$) denote the average distribution of messages passed from a checknode to an information bitnode (respectively, parity bitnode), at iteration ℓ.

Under the cycle-free condition, $P_\ell, \widetilde{P}_\ell, Q_\ell, \widetilde{Q}_\ell$ satisfy the following recursion:

$$P_\ell = F_u \otimes \lambda(Q_\ell) \tag{10}$$
$$\widetilde{P}_\ell = F_u \otimes \widetilde{Q}_\ell \tag{11}$$
$$Q_\ell = \Gamma^{-1}\left(\Gamma(\widetilde{P}_{\ell-1})^{\otimes 2} \otimes \Gamma(P_{\ell-1})^{\otimes(a-1)}\right) \tag{12}$$
$$\widetilde{Q}_\ell = \Gamma^{-1}\left(\Gamma(\widetilde{P}_{\ell-1}) \otimes \Gamma(P_{\ell-1})^{\otimes a}\right) \tag{13}$$

for $\ell = 1, 2, \ldots$, with initial condition $P_0 = \widetilde{P}_0 = \Delta_0$, where F_u denotes the distribution of the channel observation messages (4), $\otimes$ denotes convolution of distributions, defined by

$$(F \otimes G)(z) = \int F(z-t)dG(t) \tag{14}$$

where $^{\otimes m}$ denotes m-fold convolution,

$$\lambda(F) \triangleq \sum_{i=2}^{d} \lambda_i F^{\otimes(i-1)},$$

$\Gamma(F_x)$ is the distribution of $y = \gamma(x)$ (defined on $\mathbb{F}_2 \times \mathbb{R}_2$), when $x \sim F_x$, and Γ^{-1} denotes the inverse mapping of Γ, i.e., $\Gamma^{-1}(G_y)$ is the distribution of $x = \gamma^{-1}(y)$ when $y \sim G_y$. $\square$

Thus, the sequences of (10)–(13) are sequences of pairs of symmetric distributions $(P_\ell, \widetilde{P}_\ell)$). For this system, the BER at iteration ℓ is given by $\mathrm{Pe}(P_\ell)$.

It is easy to see that $(\Delta_\infty, \Delta_\infty)$ is a fixed point of (10)–(13). The local stability of this fixed point is given by the following result.

Theorem 1: The fixed point $(\Delta_\infty, \Delta_\infty)$ for the DE is locally stable if and only if

$$\lambda_2 < \frac{e^r(e^r - 1)}{a + 1 + e^r(a-1)} \tag{15}$$

where $r = -\log(\int e^{-z/2}dF_u(z))$.

Proof: See Appendix I. $\square$

Here necessity and sufficiency are used in the sense of [11]. By following steps analogous to [11], it can be shown that if (15) holds, then there exists $\xi > 0$ such that if for some $\ell \in \mathbb{N}$

$$\mathrm{Pe}(RP_\ell(P_0, \widetilde{P}_0) + (1-R)\widetilde{P}_\ell(P_0, \widetilde{P}_0)) < \xi$$

then $\mathrm{Pe}(RP_\ell + (1-R)\widetilde{P}_\ell)$ converges to zero as ℓ tends to infinity. On the contrary, if λ_2 is strictly larger than the right-hand side (RHS) of (15), then there exists $\xi > 0$ such that for all $\ell \in \mathbb{N}$

$$\mathrm{Pe}(RP_\ell(P_0, \widetilde{P}_0) + (1-R)\widetilde{P}_\ell(P_0, \widetilde{P}_0)) > \xi.$$

III. IRA Ensemble Optimization

In this section, we tackle the problem of optimizing the IRA code ensemble parameters for a broad class of binary-input symmetric-output channels.

A property of DE given in Proposition 1 is that $\mathrm{Pe}(P_\ell)$ for $\ell = 1, 2, \ldots$ is a nonincreasing nonnegative sequence. Hence, the limit $\lim_{\ell\to\infty} \mathrm{Pe}(P_\ell)$ exists. Consider a family of channels

$$\mathcal{C}(\nu) = \{p^{\nu}_{Y|X} : \nu \in \mathbb{R}_+\}$$

where the channel parameter ν is, for example, an indicator of the noise level in the channel. Following [31], we say that $\mathcal{C}(\nu)$ is monotone with respect to the IRA code ensemble $(\{\lambda_i\}, a)$ under BP decoding if, for any finite ℓ

$$\nu \le \nu' \Leftrightarrow \mathrm{Pe}(P_\ell) \le \mathrm{Pe}(P'_\ell)$$

where P_ℓ and P'_ℓ are the message distributions at iteration ℓ of DE applied to channels $p^{\nu}_{Y|X}$ and $p^{\nu'}_{Y|X}$, respectively.

Let $\mathrm{BER}(\nu) = \lim_{\ell\to\infty} \mathrm{Pe}(P_\ell)$, where $\{P_\ell\}$ is the trajectory of DE applied to the channel $p^{\nu}_{Y|X}$. The *threshold* $\nu^\star$ of the ensemble $(\{\lambda_i\}, a)$ over the monotone family $\mathcal{C}(\nu)$ is the worst case channel parameter for which the limiting BER is zero, i.e.,

$$\nu^\star = \sup\{\nu \ge 0 : \mathrm{BER}(\nu) = 0\}. \tag{16}$$

Thus, for every value of ν, the optimal IRA ensemble parameters a and $\{\lambda_i\}$ maximize R subject to vanishing $\mathrm{BER}(\nu) = 0$, i.e., are solution of the optimization problem

$$\Big\{ \text{maximize} \quad a\textstyle\sum_{i=2}^{d} \lambda_i/i$$

the solution of which can be found by some numerical techniques, as in [11]. However, the constraint $\text{BER}(\nu) = 0$ is given directly in terms of the fixed point of the DE recursion, and makes optimization very computationally intensive.

A variety of methods have been developed in order to simplify the code ensemble optimization [19], [24], [14], [32]. They consist of replacing the DE with a dynamical system defined over the reals (rather than over the space of distributions), whose trajectories and fixed points are related in some way to the trajectories and the fixed point of the DE. Essentially, all proposed approximated DE methods can be formalized as follows. Let $\Phi: \mathcal{F}_{\text{sym}} \to \mathbb{R}$ and $\Psi: \mathbb{R} \to \mathcal{F}_{\text{sym}}$ be mappings of the set of symmetric distributions to the real numbers and *vice versa*. Then, a dynamical system with state space $\mathbb{R}^2$ can be derived from (10)–(13) as

$$x_\ell = \Phi\left(F_u \otimes \lambda(\mathsf{Q}_\ell)\right) \tag{18}$$
$$\widetilde{x}_\ell = \Phi\left(F_u \otimes \widetilde{\mathsf{Q}}_\ell\right) \tag{19}$$
$$\mathsf{Q}_\ell = \Gamma^{-1}\left(\Gamma\left(\Psi(\widetilde{x}_{\ell-1})\right)^{\otimes 2} \otimes \Gamma\left(\Psi(x_{\ell-1})\right)^{\otimes(a-1)}\right) \tag{20}$$
$$\widetilde{\mathsf{Q}}_\ell = \Gamma^{-1}\left(\Gamma\left(\Psi(\widetilde{x}_{\ell-1})\right) \otimes \Gamma\left(\Psi(x_{\ell-1})\right)^{\otimes a}\right) \tag{21}$$

for $\ell = 1, 2, \ldots$, with initial condition $x_0 = \widetilde{x}_0 = \Phi(\Delta_0)$, and where $(x_\ell, \widetilde{x}_\ell)$ are the system state variables.

By eliminating the intermediate distributions Q_ℓ and $\widetilde{\mathsf{Q}}_\ell$, we can put (18)–(21) in the form

$$\begin{aligned} x_\ell &= \phi(x_{\ell-1}, \widetilde{x}_{\ell-1}) \\ \widetilde{x}_\ell &= \widetilde{\phi}(x_{\ell-1}, \widetilde{x}_{\ell-1}). \end{aligned} \tag{22}$$

For all DE approximations considered in this work, the mappings Φ and Ψ and the functions ϕ and $\widetilde{\phi}$ satisfy the following desirable properties.

1) $\Phi(\Delta_0) = 0$, $\Phi(\Delta_\infty) = 1$.
2) $\Psi(0) = \Delta_0$, $\Psi(1) = \Delta_\infty$.
3) ϕ and $\widetilde{\phi}$ are defined on $[0,1] \times [0,1]$ and have range in $[0,1]$.
4) $\phi(0,0) > 0$ and $\widetilde{\phi}(0,0) > 0$.
5) $\phi(1,1) = \widetilde{\phi}(1,1) = 1$, i.e., $(1,1)$ is a fixed point of the recursion (22). Moreover, this fixed point corresponds to the zero-BER fixed point $(\Delta_\infty, \Delta_\infty)$ of the exact DE.
6) If $F_u \neq \Delta_0$, the function $\widetilde{\phi}(x, \widetilde{x}) - \widetilde{x}$ is strictly decreasing in $\widetilde{x}$ for all $x \in [0,1]$. Therefore, the equation

$$\widetilde{x} = \widetilde{\phi}(x, \widetilde{x})$$

has a unique solution in $[0,1]$ for all $x \in [0,1]$. This solution will be denoted by $\widetilde{x}(x)$.

It follows that all fixed points of (22) must satisfy

$$x = \phi(x, \widetilde{x}(x)) \tag{23}$$

and that in order to avoid fixed points other than $(1,1)$, (23) must not have solutions in the interval $[0,1)$, i.e., it must satisfy

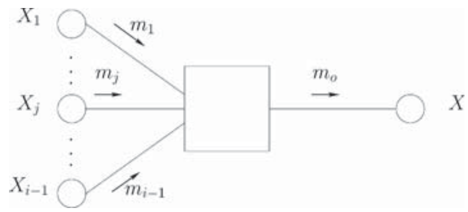

Fig. 3. EXIT model.

Notice that, in general, (24) is neither a necessary nor a sufficient condition for the uniqueness of the zero-BER fixed point of the exact DE. However, if the quality of the DE approximation is good, this provides a heuristic for the code ensemble optimization.

By replacing the constraint $\text{BER}(\nu) = 0$ by (24) in (17), we obtain the *approximated* IRA ensemble optimization method as

$$\begin{cases} \text{maximize} & a \sum_{i=2}^{d} \lambda_i / i \\ \text{subject to} & \sum_{i=2}^{d} \lambda_i = 1,\ \lambda_i \geq 0, \quad \forall\, i \\ \text{and to} & x < \phi(x, \widetilde{x}(x)), \quad \forall\, x \in [0,1). \end{cases} \tag{25}$$

Approximations of the DE recursion differ essentially in the choice of Φ and Ψ, and in the way the *intermediate* distributions Q_ℓ and $\widetilde{\mathsf{Q}}_\ell$ and the channel message distribution F_u are approximated. Next, we illustrate the approximation methods considered in this work.

A. *EXIT Functions*

Several recent works show that DE can be accurately described in terms of the evolution of the mutual information between the variables associated with the bitnodes and their messages (see [12], [33]–[35], [13], [23], [18]).

The key idea in order to approximate DE by mutual information evolution is to describe each computation node in BP decoding by a mutual information transfer function. For historical reasons, this function is usually referred to as the EXtrinsic mutual Information Transfer (EXIT) function.

EXIT functions are generally defined as follows. Consider the model of Fig. 3, where the box represents a generalized computation node of the BP algorithm (i.e., it might contain a subgraph formed by several nodes and edges, and might depend on some other random variables such as channel observations, not shown in Fig. 3). Let $m_1, \ldots, m_{i-1}$ denote the input messages, assumed independent and identically distributed (i.i.d.) $\sim F_{\text{in}}$, and let $m_o \sim F_{\text{out}}$ denote the output message. Let X_j denote the binary code symbol associated with message m_j, for $j = 1, \ldots, i-1$, and let X denote the binary code symbol associated with message m_o. Since $F_{\text{in}}, F_{\text{out}} \in \mathcal{F}_{\text{sym}}$, we can think of m_j and m_o as the outputs of binary-input symmetric-output channels with inputs X_j and X and transition probabilities

$$P(m_j \leq z | X_j = 0) = F_{\text{in}}(z) \tag{26}$$
$$P(m_o \leq z | X = 0) = F_{\text{out}}(z) \tag{27}$$

respectively.

The channel (26) models the *a priori* information that the node receives about the symbols X_j's, and the channel (27)

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.