Finite Alphabet Iterative Decoders Robust to Faulty Hardware: Analysis and Selection

Elsa DUPRAZ[1], David DECLERCQ[1], Bane VASIĆ[2] and Valentin SAVIN[3]

[1] ETIS - ENSEA / Univ. Cergy-Pontoise / CNRS UMR-8051, Cergy-Pontoise, France, {dupraz,declercq}@ensea.fr
[2] Dept. of Electrical and Computer Eng., University of Arizona, Tucson, AZ 85721, USA, vasic@ece.arizona.edu
[3] CEA-LETI, Minatec Campus, Grenoble, France, valentin.savin@cea.fr

***Abstract*—In this paper, we analyze Finite Alphabet Iterative Decoders (FAIDs) running on faulty hardware. Under symmetric error models at the message level, we derive the noisy Density Evolution equations, and introduce a new noisy threshold phenomenon (called functional threshold), which accurately characterizes the convergence behavior of LDPC code ensembles under noisy-FAID decoding. The proposed functional threshold is then used to identify FAIDs which are particularly robust to the transient hardware noise. Finite-length simulations are drawn to verify the validity of the asymptotical study.**

I. INTRODUCTION

Nowadays, reliability is becoming a major issue in the design of electronic devices. On one hand, a huge increase in the integration factors coupled with important reduction of the chip sizes makes electronic devices much more sensitive to noise and may induce transient errors on operations performed by the circuit. On the other hand, the involved delicate fabrication process make electronic components more prone to defects and may also cause permanent errors in the computation. As a consequence, in the context of communication and storage, errors may not only come from the transmission channels, but also from the faulty hardware.

The problem of error correction performed on faulty hardware was early addressed by Taylor [9] and Kuznetsov [2] who were the first to analyze the correction capability of Low Density Parity Check (LDPC) decoders made of unreliable components. More recently, the performance of noisy hard-decision Gallager-B decoders was investigated, both for binary [11] and non-binary [7] alphabets. Furthermore, [10] introduced a framework for the performance analysis of noisy LDPC decoders in terms of asymptotic error probability referred to as noisy Density Evolution (noisy-DE), and considered the case of Gallager-A and infinite-precision Belief Propagation decoders. From the same noisy-DE framework, [3] proposed an analysis of the behavior of discrete min-sum decoders, for which the exchanged messages are no longer binary but are quantized soft information stored in a finite (and typically small) number of bits. Except [1] which deals with both transient and permanent errors, most of these works consider only transient errors, as we will do in this paper.

Recently, a new class of LDPC decoders referred to as Finite Alphabet Iterative Decoders (FAIDs) has been introduced [8]. In these decoders, the messages take their values in small alphabets and the variable node update is derived through a predefined mapping that has to satisfy some particular properties. For decoding on faulty hardware, the FAID framework offers the possibility to define a large collection of these mappings, each corresponding to a particular decoding algorithm, with potentially different behaviors in terms of tolerance to transient errors. In this paper, we propose a method for the selection of decoders robust to transient errors introduced by the faulty hardware. This selection procedure is based on an asymptotic performance analysis of noisy-FAIDs realized with noisy-DE. In particular, we introduce a noisy-DE threshold definition different from the useful threshold defined in [10]. This definition characterizes more accurately the convergence behavior of LDPC code ensembles under noisy-FAIDs decoding.

The remainder of the paper is organized as follows. The FAID framework is first described, and we show how it enables to define a large collection of decoders with different properties. Then, the error models considered for the faulty hardware are introduced. Next, the noisy-DE equations are derived and, from these equations, we introduce the threshold definition and explain how to analyze the asymptotic performance of the decoders. In a final part of the paper, we propose a noisy-DE based framework to identify, within the diversity of a large number of FAIDs, the decoders that are naturally more robust to errors introduced by the hardware. Finite-length simulation results illustrate the gain in performance at considering robust FAIDs on faulty hardware.

II. FINITE ALPHABET ITERATIVE DECODERS RUNNING ON FAULTY HARDWARE

In the following, we assume that the transmission channel is a Binary Symmetric Channel (BSC) with parameter α.

A. FAID Update Rules

An N_s-level FAID is defined as a 4-tuple given by $\mathrm{D} = (\mathcal{M}, \mathcal{Y}, \Phi_v, \Phi_c)$. The message alphabet is finite and can be defined as $\mathcal{M} = \{-L_s, \dots, -L_1, 0, L_1, \dots, L_s\}$, where $L_i \in \mathbb{R}^+$ and $L_i > L_j$ for any $i > j$. It thus consists of $N_s = 2s + 1$ levels to which the message values belong. For the BSC, the set $\mathcal{Y}$, which denotes the set of possible channel values, is defined as $\mathcal{Y} = \{\pm \mathrm{B}\}$, where $\mathrm{B} \in \{-L_s, \dots, L_s\}$. For the n-th symbol of the codeword, the channel value $y_n \in \mathcal{Y}$ corresponding to node v_n is determined based on its received value. Here, we use the mapping $0 \rightarrow \mathrm{B}$ and $1 \rightarrow -\mathrm{B}$. In the following, $\mu_1, \ldots, \mu_{d_c-1}$ denote the extrinsic incoming messages to a Check Node (CN) of degree d_c and let $\eta_1, \ldots, \eta_{d_v-1}$ be the extrinsic incoming messages to a Variable Node (VN) of degree d_v.

At each iteration of the iterative decoding process, the following operations defined in [8] are performed on the messages. The Check Node Update (CNU) function $\Phi_c : \mathcal{M}^{d_c-1} \rightarrow \mathcal{M}$ used for the update at a Check Node (CN) of degree d_c is given by $\Phi_c(\mu_1, \ldots, \mu_{d_c-1})$ and corresponds to the CNU of the standard min-sum decoding. The Variable Node Update (VNU) function $\Phi_v : \mathcal{M}^{d_v-1} \times \mathcal{Y} \rightarrow \mathcal{M}$ used for the update at a Variable Node (VN) v_n, $n = 0 \ldots N-1$ of degree d_v, is given by $\Phi_v(\eta_1, \cdots, \eta_{d_v-1}, y_n)$. The properties that Φ_v must verify are given in [8]. To finish, at the end of each decoding iteration, the *A Posteriori* (APP) computation produces messages γ calculated from the function $\Phi_a : \mathcal{M}^{d_v} \times \mathcal{Y} \rightarrow \bar{\mathcal{M}}$, where $\bar{\mathcal{M}} = \{-L_{s'}, \ldots, L_{s'}\}$ and $s' = 2s+1$. The function Φ_a is expressed as

$$\Phi_a(\eta_1, \ldots, \eta_{d_v}, y_n) = \sum_{j=1}^{d_v} \eta_j + y_n \quad . \qquad (1)$$

It is computed on a bigger alphabet $\bar{\mathcal{M}}$ in order to limit the influence of saturation effects when calculating the sum. The hard-decision bit corresponding to each variable node v_n is given by the sign of the APP computation. If the output of Φ_a is 0, then the hard-decision bit is selected at random and takes value 0 with probability $1/2$.

The VNU Φ_v can also be represented as a Look-Up Table (LUT) that is defined for a specific channel value. Table I shows an example of LUT for a 7-level FAID and column-weight three codes when the channel value is $-\mathrm{B}$. The corresponding LUT for the value $+\mathrm{B}$ can be deduced by symmetry. In fact, the VNU formulation defines a large collection of mappings with common characteristics but potentially different abilities to be robust to noise in the decoder. That is why, after introducing the considered errors models for the faulty hardware, we propose a method to analyze the asymptotic performance of noisy-FAIDs. From this method, we will be able to compare the performance of the decoder for different choices of Φ_v and thus to identify the VNU that are robust to faulty hardware.

Table I: LUT $\Phi_v^{(\mathrm{opt})}$ reported in [8] optimized for the error floor

$\boldsymbol{m_1/m_2}$	$\boldsymbol{-L_3}$	$\boldsymbol{-L_2}$	$\boldsymbol{-L_1}$	$\boldsymbol{0}$	$\boldsymbol{+L_1}$	$\boldsymbol{+L_2}$	$\boldsymbol{+L_3}$
$\boldsymbol{-L_3}$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_1$
$\boldsymbol{-L_2}$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$	L_1
$\boldsymbol{-L_1}$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	$-L_1$	$-L_1$	L_1
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	0	L_1
$\boldsymbol{L_1}$	$-L_3$	$-L_2$	$-L_1$	0	0	L_1	L_2
$\boldsymbol{L_2}$	$-L_3$	$-L_1$	$-L_1$	0	L_1	L_1	L_3
$\boldsymbol{L_3}$	$-L_1$	L_1	L_1	L_1	L_2	L_3	L_3

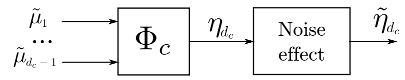

Fig. 1: Function decomposition for the CNU

B. Error Models for the Faulty Hardware

In a first step of our analysis of the faulty hardware, we assume that the noise is introduced at a message level and appears only at the output of the function computation. More precisely, we assume that the noisy function can be decomposed as a noiseless function followed by the noise effect (see Figure 1 for the case of the CNU). As a consequence, η and μ represent the messages at the output of the noiseless CNU Φ_c and VNU Φ_v respectively, and their noisy versions are denoted $\tilde{\eta}$, $\tilde{\mu}$. The noise effect for the messages at the output of Φ_c can be described by a probability transition matrix M_c such that $\forall k, m \in \{-L_s, \ldots, L_s\}$,

$$\mathrm{prob}(\tilde{\eta} = m | \eta = k) = M_c(k, m). \qquad (2)$$

Note that, here, with an abuse of notation, the coefficients of the matrix are indexed with the values $-L_s, \ldots, 0, \ldots, +L_s$. This convention will be used for all the vectors and matrices introduced in the remaining of the paper.

In this paper, we consider noise models with a Sign-Preserving (SP) property. By doing this, we in fact assume that the noise is only on the amplitude on the messages, and not on their sign. Such a model allows to study the robustness of the decoders to errors on the reliability of the estimated value associated to each VN, and not on the estimated value itself. However, it assumes extra-protection at the hardware level in computing the sign. M_c is defined as

$$\mathrm{M}_c = \begin{bmatrix} 1-p_c & \frac{p_c}{s} & \frac{p_c}{s} & \frac{p_c}{s} & 0 & 0 & 0 \\ \frac{p_c}{s} & 1-p_c & \frac{p_c}{s} & \frac{p_c}{s} & 0 & 0 & 0 \\ \frac{p_c}{s} & \frac{p_c}{s} & 1-p_c & \frac{p_c}{s} & 0 & 0 & 0 \\ \frac{p_c}{2s} & \frac{p_c}{2s} & \frac{p_c}{2s} & 1-p_c & \frac{p_c}{2s} & \frac{p_c}{2s} & \frac{p_c}{2s} \\ 0 & 0 & 0 & \frac{p_c}{s} & 1-p_c & \frac{p_c}{s} & \frac{p_c}{s} \\ 0 & 0 & 0 & \frac{p_c}{s} & \frac{p_c}{s} & 1-p_c & \frac{p_c}{s} \\ 0 & 0 & 0 & \frac{p_c}{s} & \frac{p_c}{s} & \frac{p_c}{s} & 1-p_c \end{bmatrix}$$

where $0 \leq p_c \leq 1$. The probability transition matrix for the noise effect at the output of the VNU computation is denoted M_v and is given by replacing p_c by p_v. The resulting noisy messages $\tilde{\mu}$, $\tilde{\eta}$ are then the inputs of the message updates Φ_c, Φ_v, Φ_a. The noise effect at the end of the APP computation (1) is represented by the probability transition matrix M_a obtained from M_v by adapting the size of the matrix to the alphabet $\bar{\mathcal{M}}$ and replacing s by s'. The parameter p_v is used for the APP computation because this operation is usually performed at the VN part of the decoding.

As in the following the performance analysis is realized with noisy-DE, only *symmetric* models [10] are considered so that the simplifying all-zero codeword assumption can be performed. Furthermore, the noise is assumed to be only at the output of the noiseless functions. A perhaps more relevant model could be to consider some noise effect introduced *inside* the functions, for example during elementary operations such as the minimum computation between two elements in Φ_c, as in [3]. However, although the models introduced here may not capture all the noise effects, they enable to analyze the behavior and robustness of noisy decoders without having to consider a particular and fixed hardware implementation. More accurate faulty hardware models will be considered in future works.

III. Density Evolution of Noisy Message Passing Decoding

This section presents the noisy-DE framework for the asymptotic analysis of FAIDs on faulty hardware. DE [10] consists of expressing the probability mass function (pmf) of the messages at successive iterations under the local independence assumption, that is the assumption that the messages arriving at a node are independent. The DE analysis enables to characterize the asymptotic behavior of the decoding algorithm under particular decoder noise conditions for a given VNU Φ_v and is valid on average over all possible LDPC code constructions, when infinite length LDPC graphs are considered. Note that here, not only the considered channel model, but also the noiseless functions and the decoding noise models are symmetric in the sense of [10]. Thus the final error probability of the decoder does not depend on the transmitted codeword.

In the following, we first give the expression of the pmf of the messages at successive iterations and then explain how they can be used to characterize the asymptotic behavior of the decoders. Note that the presented analysis holds for regular LDPC codes. However, the following expressions can be easily generalized to the case of irregular codes.

A. Noisy Density Evolution Recursion

Denote $\mathbf{q}_0$ the pmf in vector form of the initial messages. The k-th component $q_0(k)$ of $\mathbf{q}_0$ is the probability that the initial message takes value $k \in \{-L_s, \dots, L_s\}$. Correspondingly, denote $\mathbf{q}^{(\ell)}$, $\mathbf{r}^{(\ell)}$ the vector forms of the pmfs of the messages at the outputs of the VNU and the CNU at iteration ℓ. Their noisy versions are denoted $\tilde{\mathbf{q}}^{(\ell)}$ and $\tilde{\mathbf{r}}^{(\ell)}$, respectively.

The density evolution is initialized with the pmf of the channel value, that is

$$q_0(-B) = 1-\alpha \quad q_0(+B) = \alpha \quad q_0(k) = 0 \text{ elsewhere.}$$

The pmf $\mathbf{r}^{(\ell)}$ of the output of the CNU is obtained from the expression of Φ_c as

$$r^{(\ell)}(\eta_{d_v}) = \sum_{\substack{(\mu_1,\dots,\mu_{d_c-1}):\\ \Phi_c(\mu_1,\dots,\mu_{d_c-1})=\eta_{d_v}}} \prod_{i=1}^{d_c-1} \tilde{q}^{(\ell-1)}(\mu_i) \tag{3}$$

The noisy pmf is then obtained directly in vector form as

$$\tilde{\mathbf{r}}^{(\ell)} = \mathbf{M}_c \mathbf{r}^{(\ell)}. \tag{4}$$

We proceed the same way to obtain the DE equations for the VNU and we get

$$\begin{aligned} q^{(\ell)}(\mu_{d_c}) = & \sum_{\substack{(\eta_1,\dots,\eta_{d_v-1}):\\ \Phi_v(\eta_1,\dots,\eta_{d_v-1},-B)=\mu_{d_c}}} q_0(-B) \prod_{i=1}^{d_v-1} \tilde{r}^{(\ell)}(\eta_i) \\ & + \sum_{\substack{(\eta_1,\dots,\eta_{d_v-1}):\\ \Phi_v(\eta_1,\dots,\eta_{d_v-1},+B)=\mu_{d_c}}} q_0(+B) \prod_{i=1}^{d_v-1} \tilde{r}^{(\ell)}(\eta_i) \end{aligned} \tag{5}$$

and

$$\tilde{\mathbf{q}}^{(\ell)} = \mathbf{M}_v \mathbf{q}^{(\ell)}. \tag{6}$$

Finally, applying recursively the sequence of 4 equations (3), (4), (5) and (6) implements one recursion of the Noisy Density Evolution for FAIDs over the BSC channel. Note that, to avoid complexity explosion when the VN and CN degrees increase, (3) and (5) can be computed recursively on the inputs, as in [4].

From the recursion, we now want to determine the error probability of the decoder. This can be obtained from the pmf of the APP computation. Denote $\mathbf{q}_{\text{app}}^{(\ell)}$ and $\tilde{\mathbf{q}}_{\text{app}}^{(\ell)}$ the respective noiseless and noisy pmfs of the messages at the output of the APP computation. They can be expressed from (1) as

$$\begin{aligned} q_{\text{app}}^{(\ell)}(\gamma) = & \sum_{\substack{(\eta_1,\dots,\eta_{d_v}):\\ \Phi_a(\eta_1,\dots,\eta_{d_v},-B)=\gamma}} q_0(-B) \prod_{i=1}^{d_v} \tilde{r}^{(\ell)}(\eta_i) \\ & + \sum_{\substack{(\eta_1,\dots,\eta_{d_v}):\\ \Phi_a(\eta_1,\dots,\eta_{d_v},+B)=\gamma}} q_0(+B) \prod_{i=1}^{d_v} \tilde{r}^{(\ell)}(\eta_i) \end{aligned} \tag{7}$$

and

$$\tilde{\mathbf{q}}_{\text{app}}^{(\ell)} = \mathbf{M}_a \mathbf{q}_{\text{app}}^{(\ell)}. \tag{8}$$

Finally, the error probability at each iteration can be computed under the all-zero codeword assumption as

$$\tilde{P}_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v) = \frac{1}{2}\tilde{q}_{\text{app}}^{(\ell)}(0) + \sum_{\gamma<0} \tilde{q}_{\text{app}}^{(\ell)}(\gamma). \tag{9}$$

It thus suffices to study the asymptotic error probability for a given Φ_v, that is the limit of $\tilde{P}_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v)$ when ℓ goes to infinity, to characterize the asymptotic behavior of the noisy decoder.

Denote $P_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v)$ the error probability assuming that the APP computation is error free, and denote $P_e^{(\text{lb})}(p_v) = \frac{1}{2s'}p_v$. We show that the following inequalities hold.

Proposition 1. *For the SP error model, the two following inequalities hold at every iteration ℓ*

1) $\tilde{P}_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v) \geq P_e^{(lb)}(p_v)$, *and*
2) $\tilde{P}_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v) \geq P_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v)$.

Because of the lack of space, the proofs of the propositions are not in the paper.

The first lower bound is achieved when the error is only on the APP computation, assuming that the VNU and CNU computations are error-free and able to correct all the errors from the channel. On the opposite, to obtain the

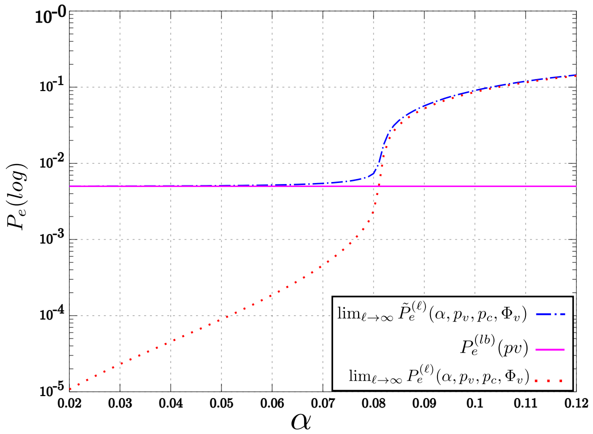

Fig. 2: Asymptotic error probabilities and lower bound for $p_v = 3 \times 10^{-2}, p_c = 10^{-3}$

second inequality, we assume that the VNU and CNU can be in error but that the APP computation is error-free. In the following, we use the two inequalities to characterize the convergence behavior of the decoder.

B. Analysis of Convergence Behaviors

For noiseless decoders, the maximum channel parameter α such that $\lim_{\ell\to+\infty} P_e^{(\ell)}(\alpha, 0, 0, \Phi_v) = 0$ is called the *threshold* of the code [5]. However, from Proposition 1, we see that this condition cannot be reached for noisy-FAIDs. As a consequence, there is a need to introduce another threshold definition that identifies the channel parameters for which the decoder can correct the maximum possible number of errors from the channel.

For noisy-DE, the author in [10] defines the *useful* threshold for given decoding noise conditions (p_v, p_c) as the maximum parameter α such that $\lim_{\ell\to+\infty} \tilde{P}_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v) \le \alpha$. The useful threshold indicates what are the faulty hardware conditions and the maximum channel noise that a noisy decoder can tolerate to *reduce* the level of noise. In this paper, we introduce another threshold definition, which relies on more stringent convergence conditions of the noisy DE recursion. The threshold, that we call functional threshold, is defined as follows.

Definition 1. *For fixed values p_v, p_c, and a given FAID Φ_v, the* functional threshold $\overline{\alpha}(p_v, p_c, \Phi_v)$ *is defined as*

$$\overline{\alpha}(p_v, p_c, \Phi_v) = \arg\max_{\alpha} \left\{ \lim_{\ell\to+\infty} P_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v) \textit{ exists} \right.$$
$$\left. \textit{and} \lim_{\ell\to+\infty} P_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v) \le P_e^{(lb)}(p_v) \right\}.$$

The existence condition is required because from [3] the error probability sometimes does not converge for some particular decoders and noise conditions. We define the functional threshold as the value of α for which $\lim_{\ell\to+\infty} P_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v)$ crosses the lower bound $P_e^{(\mathrm{lb})}(p_v)$ (see Figure 2). The useful threshold cannot predict well the asymptotic behavior of the decoders in the sense that it cannot identify the channel parameters leading to an asymptotic performance close to the lower bound $P_e^{(\mathrm{lb})}(p_v)$. In addition, using only $P_e^{(\mathrm{lb})}(p_v)$ as a reference to define the threshold is not sufficient because, in general, the lower bound is not achieved and the gap between the asymptotic error probability and $P_e^{(\mathrm{lb})}(p_v)$ slightly increases with the value of α. Our definition of the functional threshold enables to determine the set of parameters α for which $\tilde{P}_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v)$ is close enough to the lower bound. Furthermore, this condition leads to the following result.

Proposition 2. *If there exists a value $\alpha^\star$ such that $\lim_{\ell\to+\infty} P_e^{(\ell)}(\alpha^\star, p_v, p_c, \Phi_v) = P_e^{(lb)}(p_v)$, then the following inequality holds*

$$2P_e^{(lb)}(p_v) - \frac{p_v^2}{2s^2} \le \lim_{\ell\to+\infty} \tilde{P}_e^{(\ell)}(\alpha^\star, p_v, p_c, \Phi_v) \le 2P_e^{(lb)}(p_v). \tag{10}$$

The term $\frac{p_v^2}{2s^2}$ in (10) is actually small compared to the value $P_e^{(\mathrm{lb})}(p_v) = \frac{p_v}{2s}$. Therefore, for the channel parameter α for which $P_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v)$ crosses $P_e^{(\mathrm{lb})}(p_v)$, the asymptotic probability $\tilde{P}_e^{(\ell)}(\alpha, p_v, p_c, \Phi_v)$ gets very close to $2P_e^{(\mathrm{lb})}(p_v)$.

From this proposition, we see that the introduced threshold definition enables to determine the set of channel parameters that leads to an asymptotic error probability that is close to the lower bound. This threshold definition gives a criterion for the prediction and the comparison of the performance of different decoders under particular faulty hardware conditions. The next section thus presents the selection process of robust FAIDs based on this criterion.

IV. Selection of FAIDs Robust to Faulty Hardware

In this paper, we want to capitalize on the diversity of FAID update rules Φ_v and behavior to identify if there are iterative decoders which are naturally more robust than others under faulty hardware implementation.

From [8, Theorem 1], we know that even by restricting the message alphabet size to $N_s = 7$, there are $530\,803\,988$ different FAIDs, which is too large for a systematic analysis. Instead, we rely on previous work on FAID, and start with a collection of $N_D = 5291$ FAIDs which correspond to column-weight tree codes and have been selected from the analysis on trapping sets presented in [8]. As a result of this selection process, all of these N_D FAIDs have both good noiseless-DE thresholds, and good performance in the error floor. We now conduct a noisy-DE analysis on this set of N_D FAIDs by computing, for each of them, the values of their functional threshold. As an example, Figure 3 represents the noisy versus noiseless thresholds obtained for all the FAIDs with $p_v = 0.05$, $p_c = 0.05$. We see that although the considered decoders have all good noiseless threshold, they exhibit very different behaviors when the decoder is noisy.

Two decoders are extracted from the set of N_D FAIDs. The first one denoted $\Phi_v^{\text{(robust)}}$ is the decoder that *minimizes* the difference between noiseless and noisy threshold. The second one $\Phi_v^{\text{(non-robust)}}$ is selected to *maximize* the difference between noiseless and noisy threshold. The LUTs of $\Phi_v^{\text{(robust)}}$ and $\Phi_v^{\text{(non-robust)}}$ selected for $p_v = 0.05$, $p_c = 0.05$ are given respectively in Tables II and III.

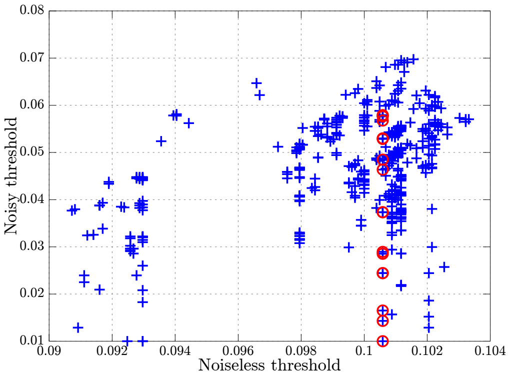

Fig. 3: Threshold for the noiseless decoder vs threshold for $p_v = 0.05$, $p_c = 0.05$ for 500 of the decoders. The circled points all have noiseless threshold value close to 0.1005

Table II: FAID rule $\Phi_v^{(\text{robust})}$

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	0
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	L_1
$-L_1$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	$-L_1$	$-L_1$	L_1
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	$-L_1$	0	L_1
$+L_1$	$-L_3$	$-L_2$	$-L_1$	$-L_1$	0	L_1	L_2
$+L_2$	$-L_2$	$-L_2$	$-L_1$	0	L_1	L_2	L_2
$+L_3$	0	L_1	L_1	L_1	L_2	L_2	L_3

V. Finite Length Simulations Results

This section gives finite-length simulation results with the noisy FAIDs that have been identified by the noisy DE analysis. Our main purpose is to compare $\Phi_v^{(\text{robust})}$, $\Phi_v^{(\text{non-robust})}$, and $\Phi_v^{(\text{opt})}$ (Table I), optimized in [8] for noiseless decoding with low error floor. The number of iterations is set to 100. We fix $p_v = p_c = 0.05$ and compare on Figure 4 the three defined decoders on the $(155, 93)$ Tanner code given in [6] with degrees $(d_v = 3, d_c = 5)$.

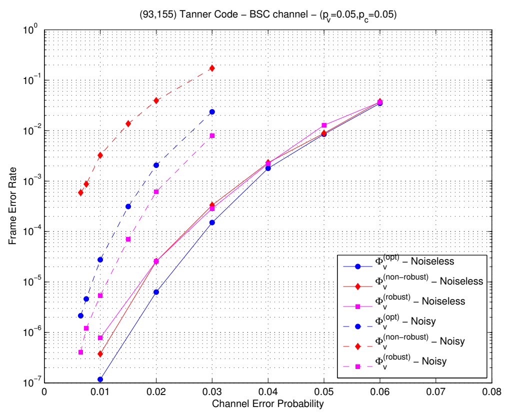

Fig. 4: Performance of Noiseless and Noisy FAIDs on the $(93, 155)$ Tanner code, with $(d_v = 3, d_c = 5)$ and $(p_v = 0.05, p_c = 0.05)$.

For the noiseless curves, as $\Phi_v^{(\text{opt})}$ has been optimized for low error floor, it performs better, as expected, than the two other FAIDs. But as $\Phi_v^{(\text{robust})}$ and $\Phi_v^{(\text{non-robust})}$ belong to a pre-determined set of good FAID decoders, they also have reasonable performance in the noiseless case. Now, from the noisy curves, we see that the results are in compliance

Table III: FAID rule $\Phi_v^{(\text{non-robust})}$ not robust to faulty Hardware

m_1/m_2	$-L_3$	$-L_2$	$-L_1$	0	$+L_1$	$+L_2$	$+L_3$
$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	0
$-L_2$	$-L_3$	$-L_3$	$-L_3$	$-L_3$	$-L_2$	0	L_2
$-L_1$	$-L_3$	$-L_3$	$-L_2$	$-L_2$	$-L_1$	0	L_2
0	$-L_3$	$-L_3$	$-L_2$	$-L_1$	0	L_1	L_3
$+L_1$	$-L_3$	$-L_2$	$-L_1$	0	0	L_1	L_3
$+L_2$	$-L_3$	0	0	L_1	L_1	L_1	L_3
$+L_3$	0	L_2	L_2	L_3	L_3	L_3	L_3

with the conclusions from the noisy functional thresholds analysis. Indeed,when the decoder is noisy, $\Phi_v^{(\text{robust})}$ performs better than $\Phi_v^{(\text{opt})}$ while $\Phi_v^{(\text{non-robust})}$ has an important loss in performance compared to the two other decoders.

VI. Conclusion

In this paper, we performed an analysis of asymptotic performance of noisy FAIDs using noisy-DE. We introduced the functional threshold that enables to predict the asymptotic behavior of noisy FAIDs. From this asymptotic analysis, we were able to identify robust FAIDs, as confirmed by the finite-length simulations.

Future works will be dedicated to the analysis of more accurate models for the faulty hardware. In particular, models that are not sign-preserving, not symmetric, and that intervene at a boolean level inside the function computation may be considered.

Acknowlegement

This work was funded in part by the Seventh Framework Programme of the European Union, under Grant Agreement number 309129 (i-RISC project), and in part by the NSF under grants CCF-0963726 and CCF-1314147.

References

[1] C-H. Huang, Y. Li, and L. Dolecek. Gallager B LDPC decoder with transient and permanent errors. In *IEEE Int. Symp. on Inf. Th. Proc.*, pages 3010–3014. IEEE, 2013.

[2] A.V. Kuznetsov. Information storage in a memory assembled from unreliable components. *Pr. Per. Inf.*, 9(3):100–114, 1973.

[3] C. Kameni Ngassa, V. Savin, and D. Declercq. Unconventional behavior of the noisy min-sum decoder over the binary symmetric channel. In *Inf. Th. and App. Workshop*, pages 1–10, 2014.

[4] C.K. Ngassa, V. Savin, and D. Declercq. Min-Sum-based decoders running on noisy hardware. In *Proc. Glob. Commun. Conf.*, 2013.

[5] T.J. Richardson and R.L. Urbanke. The capacity of Low-Density Parity-Check codes under message-passing decoding. *IEEE Trans. on Inf. Th.*, 47(2):599–618, 2001.

[6] A. Sridharan T.E. Fuja R.M. Tanner, D. Sridhara and D.J. Costello Jr. LDPC block and convolutional codes based on circulant matrices. *IEEE Trans. on Inf. Th.*, 50(12):2966–2984, 2004.

[7] H. Cho S. Yazdi and L. Dolecek. Gallager B decoder on noisy hardware. *IEEE Trans. on Comm.*, 61(5):1660–1673, 2013.

[8] L. Danjean S.K. Planjery, D. Declercq and B. Vasic. Finite alphabet iterative decoders-part i: Decoding beyond belief propagation on the binary symmetric channel. *IEEE Trans. on Comm.*, 61(10):4033–4045, 2013.

[9] M.G. Taylor. Reliable information storage in memories designed from unreliable components. *Bell Syst. Tech. Jour.*, 47(10):2299–2337, 1968.

[10] L.R. Varshney. Performance of LDPC codes under faulty iterative decoding. *IEEE Trans. on Inf. Th.*, 57(7):4427–4444, 2011.

[11] B. Vasic and S.K. Chilappagari. An information theoretical framework for analysis and design of nanoscale fault-tolerant memories based on low-density parity-check codes. *IEEE Trans. on Circ. and Syst. I: Reg. Papers*, 54(11):2438–2446, 2007.