Analysis of Taylor-Kuznetsov Memory using One-Step Majority Logic Decoder

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

[1] ETIS - ENSEA / Univ. Cergy-Pontoise / CNRS UMR-8051, Cergy-Pontoise, France
[2] Dept. of Electrical and Computer Eng., University of Arizona, Tucson, AZ 85721, USA

***Abstract*—This paper addresses the problem of constructing reliable memories from unreliable components. We consider the memory construction proposed by Taylor in which a codeword stored in a faulty memory is regularly updated by an LDPC decoder to overcome the memory degradation. We assume that the LDPC decoder used in the system is a faulty one-step majority logic decoder. Compared to [1], [2] which analyze only the faulty one-step majority logic decoder, we analyze here the reliability of the whole memory construction. We introduce a sequence of output errors probabilities at successive time instants and determine the properties and the fixed points of the sequence. From the fixed-point analysis, we define a threshold that predicts the noise level which can be tolerated for the memory to stay reliable. We finally represent the reliability regions of the Taylor-Kuznetsov memory with respect to the decoder noise parameters and validate the theoretical results with Monte-Carlo simulations.**

I. INTRODUCTION

Over the past few years, important electronic chip size reductions coupled with huge increase in integration factors have made electronic devices much more sensitive to noise. The hardware noise may introduce errors during elementary computation operations and may also affect the memory units. As a consequence, there is a need to address the issue of constructing reliable memories running on faulty hardware.

Taylor [3] and Kuznetsov [4] were the first to address the issue of constructing reliable memories built from unreliable components. In the memory architecture proposed in [3], [4], the information is stored as a codeword obtained from a Low Density Parity Check (LDPC) code. The codeword is regularly passed through an LDPC decoder in order to correct the errors introduced by the faulty hardware. As the LDPC decoders runs on the same faulty hardware as the memory, it is assumed faulty as well.

More recently, Chilappagari et al. [1], [2] and Brkic et al. [5] considered the use of a faulty One-Step Majority Logic (OS-MAJ) LDPC decoder in the memory architecture proposed by Taylor and Kuznetsov. The authors of [1], [2], [5] analyzed the Bit Error Rate (BER) performance of the OS-MAJ decoder alone, but they did not evaluate the reliability of the whole memory architecture. Vasić and Chilappagari [6] identified an equivalence between a faulty Gallager B decoder and the memory architecture proposed by Taylor and Kuznetsov. However, they evaluated the reliability of the memory only through finite-length simulations. Chilappagari et al. [7] evaluated the reliability of the memory architecture by providing an analytic expression of the maximum fraction of errors that can be corrected by the faulty LDPC decoder at successive time instants. The result in [7] was aimed at providing rigorous conditions for memory reliability, based on graph expansion arguments. However, computing expansion of large graphs is a hard problem and as a consequence, the results of [7] are not convenient for the design of a reliable memory architecture.

In this paper, we consider the memory architecture of [1], [5] with a faulty OS-MAJ decoder. In this memory architecture, discrete time instants $t = 0, \ldots, T$, are considered, and a codeword obtained from an LDPC code is stored in memory at initial time instant $t = 0$. Between two time instants t and $t + 1$, the faulty hardware induces a degradation in the memory, which is represented by a memory degradation parameter α. In order to overcome the memory degradation, the codeword stored in the memory is also passed through a OS-MAJ LDPC decoder between t and $t + 1$. As the LDPC decoder runs on the same hardware as the memory, the faulty hardware introduces some noise inside the decoder. At time instant T, the information is extracted from the memory. The codeword stored in the memory at time instant T is passed through a Gallager B LDPC decoder in order to recover the information that was initially stored at time instant $t = 0$. The performance of the memory architecture can be evaluated in terms of redundancy and reliability.

The redundancy was defined in [3] as the number of noisy elements required to construct the memory architecture divided by the memory capability, that is the number of information bits stored by the memory. It is required that the redundancy does not depend on k, which induces that the complexity of the memory architecture is linear witk k, so that the memory architecture can be constructed even for large values of k. The memory is said reliable if at time instant T, the Gallager B decoder can perfectly recover the information that was stored at time instant $t = 0$.

In this paper, we propose an analytical method to analyze the reliability of the memory architecture as a function of the memory degradation level α. We first express the error probabilities in the memory at successive time instants. Then we analyse the convergence properties of the sequence of error probabilities and introduce a threshold definition that indicates the maximum degradation level that the memory can tolerate to stay reliable. This definition enables us to represent reliability regions as a set of degradation levels and decoder

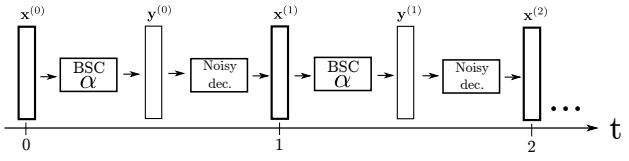

Fig. 1. Information stored in the memory at successive time instants

noise parameters for which a reliable storage of information is possible. We also provide finite-length simulation results which validate the theoretical analysis.

The outline of the paper is as follows. Section II presents the memory architecture and the faulty OS-MAJ decoder used in the architecture. Section III derives the sequence of error probabilities in the memory and analyzes the growing and convergence properties of the sequence. Section IV introduces the reliability definition and the memory threshold definition. Section IV also provides the reliability regions. Section V presents the finite-length simulation results.

II. Fault-Tolerant Memories

In this section, we present the memory architecture and the error model describing the memory degradation induced by the faulty hardware. The memory architecture and the error model were originally introduced in [3], [4] and latter considered in [1], [2], [5]–[7]. We explain how, as initially proposed by [3], LDPC codes can be used to overcome the memory degradation induced by the faulty hardware. We then describe the faulty OS-MAJ LDPC decoder used as a correction circuit.

A. Memory Degradation

Consider a memory with a storage capability of k bits, and consider the discrete time instants $t = 0, \dots, T$. Denote by $\mathbf{x}^{(0)}$ the binary information vector of length k initially stored in memory at time instant $t = 0$, and denote by $\mathbf{x}^{(t)}$ the binary information vector of length k that is in the memory at time instant t. Let $x_v^{(t)}$ be the v-th component of the vector $\mathbf{x}^{(t)}$. The memory degradation between two successive time instants t and $t+1$ is modeled by a Binary Symmetric Channel (BSC) of parameter α, which is denoted BSC(α). The BSC gives a symmetric and memoryless error model. Although such a model may not take into account all the errors induced by the faulty hardware in a realistic memory, we consider it here as a first step for the analysis.

Unfortunately, because of the memory degradation, the number of errors in $\mathbf{x}^{(t)}$ with respect to $\mathbf{x}^{(0)}$ increases with t. For large enough t, $\mathbf{x}^{(t)}$ will contain too many errors, and it will not be possible to recover the initial $\mathbf{x}^{(0)}$ from $\mathbf{x}^{(t)}$ anymore. In order to overcome this effect, the information vector is encoded by an LDPC code, as described in the following.

B. Taylor-Kuznetzov Memory Architecture

Le $\mathbf{x}^{(0)}$ be a codeword obtained from an LDPC code of dimension k defined by a parity check matrix H of size $m \times n$, with $k \leq m - n$. The vector $\mathbf{x}^{(0)}$ is stored in the memory at time instant $t = 0$, and the memory has a storage capability of k information bits. The Tanner graph of the code is composed of n Variable Nodes (VN) $v \in \{1, \dots, n\}$ and m Check Nodes (CN) $c \in \{1, \dots, m\}$. The degree of the VN v is denoted as d_v and the degree of the CN c is denoted as d_c. Here, the code is assumed to be regular, *i.e*, d_v does not depend on v, and d_c does not depend on c. Denote by $\mathcal{N}(v)$ the set of CNs connected to the VN v, and denote by $\mathcal{N}(c)$ the set of VNs connected to the CN c.

Between two successive time instants t and $t+1$, the vector $\mathbf{x}^{(t)}$ stored in the memory undergoes two operations, as depicted in Figure 1. First, $\mathbf{x}^{(t)}$ is passed through BSC(α), which gives the degraded vector $\mathbf{y}^{(t)}$. Second, $\mathbf{y}^{(t)}$ is passed through an LDPC decoder called the refresh decoder. The refresh decoder reads the memory content $\mathbf{y}^{(t)}$ at time t and outputs the vector $\mathbf{x}^{(t+1)}$ which is written back in the memory and stored in memory at instant $t+1$. The refresh decoder has to correct most of the errors introduced by the BSC between two time instant t and $t+1$, so that $\mathbf{x}^{(t)}$ is never too far away from $\mathbf{x}^{(0)}$. Here, the refresh decoder is a OS-MAJ decoder. We choose a very low complexity OS-MAJ decoder in order to limit the redundancy of the memory architecture. In addition, as the refresh decoder runs on the same faulty hardware as the memory, it will be assumed faulty as well.

Finally, when the information has to be extracted from the memory at time instant T, we allow the use of a second LDPC decoder called the final decoder. The final decoder has to reconstruct almost perfectly $\mathbf{x}^{(0)}$ from $\mathbf{x}^{(T)}$. Here, the final decoder is a noiseless Gallager B decoder. Indeed, as the final decoder is used only once at the end of the storage, we allow it to be both noiseless and more complex compared to the refresh decoder.

C. Refresh Decoder: Faulty OS-MAJ Decoder

We first describe the noiseless version of the OS-MAJ decoder. At time instant t, the refresh decoder receives the degraded vector $\mathbf{y}^{(t)}$. As a first step of the decoding, each CN c computes a message $\gamma_{c \to v}$ for all the VNs $v \in \mathcal{N}(c)$ connected to it. Let $A = \{a_1, \dots, a_d\}$ be a multiset of d binary digits and define the function realizing the XOR sum of the d terms of A as

$$f_{\oplus}(A) = a_1 \oplus \cdots \oplus a_d. \tag{1}$$

The CN messages are calculated from the function $f_{\oplus}$ as $\forall c \in \{1, \dots, m\}$, $\forall v \in \mathcal{N}(c)$,

$$\gamma_{c \to v} = f_{\oplus}\left(\{y_{v'}^{(t)}\}_{v' \in \mathcal{N}(c) \setminus v}\right). \tag{2}$$

From the CN messages it receives, each VN $v \in \{1, \dots, n\}$ computes a decision value η_v. Let $B = \{b_1, \dots, b_d\}$ be a binary multiset of size d, and let a be a binary digit. We define the majority voting function as

$$f_{\text{maj}}(B, a) = \begin{cases} 1 & \text{if } |\text{supp}(B)| > \frac{d_v}{2} \\ 0 & \text{if } |\text{supp}(B)| > \frac{d_v}{2} \\ a & \text{otherwise} \end{cases} \tag{3}$$

where $\text{supp}(B)$ is the support of B, $\bar{B}$ is the complement of B, and $|.|$ is the cardinality of a set. The decision values are calculated from the function f_{maj} as $\forall v \in \{1, \ldots, n\}$,

$$\eta_v = f_{\text{maj}}\left(\{\gamma_{c\to v}\}_{c\in\mathcal{N}(v)}, y_v^{(t)}\right). \tag{4}$$

At the end of the decoding, each VN is set as $x_v^{(t+1)} = \eta_v$. The resulting vector $\mathbf{x}^{(t+1)}$ corresponds to the vector stored in memory at time instant $t+1$.

We now describe the faulty version of the OS-MAJ decoder. Denote by $\tilde{\gamma}_{c\to v}$ the noisy versions of the CN messages $\gamma_{c\to v}$ in (2), and by $\tilde{\eta}_v$ the noisy versions of the decision values η_v. In order to obtain a faulty version of the OS-MAJ decoder from the above noiseless description, we replace the noiseless functions $f_\oplus$ in (1) and f_{maj} in (3) by their noisy versions $\tilde{f}_\oplus$ and $\tilde{f}_{\text{maj}}$. in (4).

The noisy XOR sum function $\tilde{f}_\oplus$ is defined as

$$\tilde{f}_\oplus(A) = f_\oplus(a_1, \ldots, a_d) \oplus e_\oplus \tag{5}$$

where $e_\oplus$ is a random variable distributed according to the Bernoulli distribution with parameter $p_\oplus$. The parameter $p_\oplus$ represents the error probability of the function. As for the memory degradation effect, the error model for the XOR sum function is assumed memoryless and symmetric, as a first step of the analysis. With the definition of the function $\tilde{f}_\oplus$ in (5), we assume that the noise applies only at the end of the whole XOR sum computation. Another model would be to assume that the CN computation is realized from $(d-1)$ 2-inputs faulty XOR gates, each with error probability $p_{\text{xor},2}$. However, the two models are equivalent as we have the relation $p_\oplus = P(\tilde{f}_\oplus(a_1, \ldots, a_d) \neq f_\oplus(a_1, \ldots, a_d)) = \frac{1}{2} - \frac{1}{2}(1-p_{\text{xor},2})^{(d-1)}$. It thus suffices to calculate the parameter $p_\oplus$ from $p_{\text{xor},2}$ to obtain the error model defined by the function $\tilde{f}_\oplus$ in (5).

The noisy majority voting function $\tilde{f}_{\text{maj}}$ is defined as

$$\tilde{f}_{\text{maj}}(B, a) = f_{\text{maj}}(b_1, \ldots, b_d, a) \oplus e_{\text{maj}}, \tag{6}$$

where e_{maj} is a random variable distributed according to the Bernoulli distribution with parameter p_{maj}. The parameter p_{maj} represents the error probability of the majority voting function. As for the XOR sum computation, the noise is assumed to be only at the end of function computation. While this model may not capture all the noise effects that could appear inside the majority voting function, it does not require knowledge of a particular hardware implementation of the function. However, it appears sufficient for the first step of the analysis and more accurate models will be considered in future works. Note that in the faulty OS-MAJ decoder considered in [1], [2], [5] the majority voting functions are noiseless and only the XOR gates are assumed faulty.

At the end, because of the refresh decoder, the memory constructed from faulty components requires more components than the memory built from reliable units. In order to evaluate the amount of induced additional complexity, the redundancy of the memory architecture can be evaluated as follows.

D. Redundancy of the Memory Architecture

The redundancy of the memory is defined in [3] as the number of noisy components used in the memory architecture divided by the memory capability k. For the memory architecture considered in the paper, the redundancy of the memory is expressed as [7]

$$\text{R}_{\text{ed}} = \frac{1 + D + d_v(d_c - 2)}{1 - \frac{d_v}{d_c}}. \tag{7}$$

where D is the complexity of the majority voting unit and D depends only on d_v. In (7), the final decoder is no taken into account as it is used only once at the end of the storage. The redundancy depends only on the code parameters d_v, d_c, but does not depend on the memory capability k. As a result, the complexity of the memory architecture built from unreliable components is only linear with the memory capability.

Now, we would like to determine whether and for which parameters α, $p_\oplus$, p_{maj}, the considered memory architecture is reliable. Thus for a fixed code and a given redundancy, we now analyze the reliability of the considered memory architecture with respect to the memory degradation parameter α and to the decoder noise parameters $p_\oplus$, p_{maj}.

III. Error Probability Evaluation

In this section, we analyze the reliability of the memory by expressing the bit error probabilities in the successive vectors $\mathbf{x}^{(t)}$ stored in the memory at time instants $t = 0, \ldots, T$. We first express analytically the error probability of the faulty OS-MAJ refresh decoder. We then use this probability to derive the successive error probabilities in the $\mathbf{x}^{(t)}$.

A. Error Probability of the Faulty OS-MAJ Decoder

Here, we find the error probability of the OS-MAJ decoder as a function of the memory degradation level at the input of the decoder. The error probability of the faulty OS-MAJ decoder was given in [2], [5] for different decoder noise error models. We restate it here for the error model we consider in the paper. For now, assume that the input degradation level is α, which corresponds to the degradation level in the first time interval between $t = 0$ and $t = 1$.

The BSC representing the memory degradation and the faulty functions $\tilde{f}_\oplus$ and $\tilde{f}_{\text{maj}}$ used in the decoder are symmetric in the sense of [8]. Thus, from [8], we can assume that the all-zero codeword was initially stored in memory, which greatly simplifies the analysis. From the all-zero codeword assumption, we obtain the error probability of the Majority Logic decoder by expressing the probabilities of the messages exchanged during the decoding.

Denote by $\tilde{p}_\gamma = P(\tilde{\gamma}_{c\to v} = 1)$ the probability of a noisy CN message $\tilde{\gamma}_{c\to v}$. Denote by $p_\eta = P(\eta_v = 1)$ the probability of a noiseless decision value η_v, and denote by $\tilde{p}_\eta = P(\eta_v = 1)$ the probability of its noisy version. The probability $\tilde{p}_\gamma$ of the noisy CN message $\tilde{\gamma}_{c\to v}$ can be written as

$$\tilde{p}_\gamma = \frac{1}{2} - \frac{1}{2}(1 - 2p_\oplus)(1 - \alpha)^{(d_c - 1)}. \tag{8}$$

It corresponds to the probability of an occurence of odd number of ones among the $(d_c - 1)$ inputs of the CN, or of an error in the XOR sum computation. The probability p_η of the noiseless decision value η_v is calculated depending on the parity of the VN degree d_v. If d_v is odd, p_η can be written as

$$p_\eta = \sum_{j=\lceil \frac{d_v}{2} \rceil}^{d_v} \binom{d_v}{j} \tilde{p}_\gamma^j (1-\tilde{p}_\gamma)^{(d_v-j)} \tag{9}$$

If d_v is even, we get

$$\begin{aligned} p_\eta = &\sum_{j=\frac{d_v}{2}}^{d_v} \binom{d_v}{j} \tilde{p}_\gamma^j (1-\tilde{p}_\gamma)^{(d_v-j)} \\ &- (1-\alpha)\binom{d_v}{\frac{d_v}{2}} \tilde{p}_\gamma^{\left(\frac{d_v}{2}\right)} (1-\tilde{p}_\gamma)^{\left(\frac{d_v}{2}\right)}. \end{aligned} \tag{10}$$

In both cases, p_η corresponds to the probability of a majority of 1 digits at the input of the noiseless majority voting unit.

We now express the error probability of the decoder. Denote $\nu = (p_\oplus, p_{\text{maj}})$ the decoder noise parameter pairs. The error probability $P_{e,\nu}(\alpha) = P(x_v^{(1)} = 1 | x_v^{(0)} = 0)$ of the faulty OS-MAJ decoder can be calculated from p_η as

$$P_{e,\nu}(\alpha) = p_\eta(1 - p_{\text{maj}}) + (1 - p_\eta)p_{\text{maj}}. \tag{11}$$

The error probability $P_{e,\nu}(\alpha)$ of the decoder is equal to the probability $\tilde{p}_\eta = P(\eta_v = 1)$ of the noisy decision value η_v. The error probability $P_{e,\nu}(\alpha)$ depends on the input noise level α, and on the decoder noise parameters $p_\oplus, p_{\text{maj}}$.

The expression of the error probability $P_{e,\nu}(\alpha)$ in (11) can now be used to evaluate the successive error probabilities in the memory.

B. Successive Error Probabilities in the Memory

In this section, we want to derive the expressions of the error probabilities in the $\mathbf{x}^{(t)}$ at successive time instants $t = 0, \ldots, T$. In order to do this, we also give the expressions of the degradation levels in the $\mathbf{y}^{(t)}$ at successive time instants $t = 0, \ldots, T$. As before, from the symmetry of the functions and error models considered in the memory architecture, we can assume that the all-zero codeword was initially stored in memory. According to the memory architecture defined in Section II, the successive degradation levels in the $\mathbf{y}^{(t)}$ and the successive error probabilities in the $\mathbf{x}^{(t)}$ are given in the following proposition.

Proposition 1: Denote $\beta_\nu^{(t)}(\alpha)$ the degradation level in $\mathbf{y}^{(t)}$ with respect to $\mathbf{x}^{(0)}$, *i.e.*, $\beta_\nu^{(t)}(\alpha) = P(y_v^{(t)} = 1 | x_v^{(t)} = 0)$. The successive degradation levels $\beta_\nu^{(t)}(\alpha)$ can be expressed recursively as

$$\beta_\nu^{(1)}(\alpha) = \alpha, \tag{12}$$

and $\forall t > 1$,

$$\beta_\nu^{(t)}(\alpha) = (1-\alpha) P_{e,\nu}\left(\beta_\nu^{(t-1)}(\alpha)\right) + \alpha\left(1 - P_{e,\nu}\left(\beta_\nu^{(t-1)}(\alpha)\right)\right). \tag{13}$$

The error probabilities in the successive $\mathbf{x}^{(t)}$ are given by $\delta_\nu^{(t)}(\alpha) = P(x_v^{(t)} = 1 | x_v^{(0)} = 0)$, and

$$\delta_\nu^{(t)}(\alpha) = P_{e,\nu}\left(\beta_\nu^{(t)}(\alpha)\right), \ \forall t \geq 1. \tag{14}$$

The initial $\beta_\nu^{(1)}(\alpha)$ is given by the fact that $\mathbf{y}^{(1)}$ is the output of BSC(α). At time instant $(t-1)$, the stored vector $\mathbf{x}^{(t-1)}$ has error probability $P_{e,\nu}\left(\beta_\nu^{(t-1)}(\alpha)\right)$ and is passed through BSC(α). The resulting $\mathbf{y}^{(t-1)}$ can be seen as the output of the concatenation of BSC(α) and of BSC $\left(P_{e,\nu}\left(\beta_\nu^{(t-1)}(\alpha)\right)\right)$, respectively, which gives the expression of $\beta_\nu^{(t)}(\alpha)$ in (13). The faulty decoder then produces $\mathbf{x}^{(t)}$ from its input $\mathbf{y}^{(t)}$, and as a result, the error probabilities in the successive $\mathbf{x}^{(t)}$ are given by the $\delta_\nu^{(t)}(\alpha)$ in (14).

Proposition 1 gives the recursive expression of the sequences $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ and $\{\delta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ of error probabilities in the memory. In the following, we analyze the sequence of $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ instead of the sequence of $\{\delta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$. Indeed, we are more interested in the degradation levels that the memory can tolerate than in the successive error probabilities. This is in compliance with the conventional analysis of LDPC decoders in which we define a threshold on the channel parameter.

The memory will be reliable if the successive degradation levels are small enough so that at any time instant, we can guarantee that $\mathbf{x}^{(t)}$ is in a close proximity of $\mathbf{x}^{(0)}$ and can be recovered by a perfect Gallager B decoder. In order to be able to check this condition for various values of α, $p_\oplus$, p_{maj}, and for different choices of LDPC codes, we first analyze the increasing and convergence properties of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$.

C. Sequence Properties

In this section, we first analyze the increasing properties of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$. The properties of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ are stated in the following Proposition.

Proposition 2: Consider the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ given in Proposition 1.

1) Fix $p_\oplus$ and p_{maj}. If the function $\alpha \to P_{e,\nu}(\alpha)$ is increasing with α, then
$$\forall \alpha < 1/2, \quad \beta_\nu^{(t)}(\alpha) \leq \beta_\nu^{(t+1)}(\alpha). \tag{15}$$
2) Fix $p_\oplus$ and p_{maj}. If the function $\alpha \to P_{e,\nu}(\alpha)$ is increasing with α, then
$$\forall t > 0, \quad \alpha_1 \leq \alpha_2 \ \Rightarrow \ \beta_\nu^{(t)}(\alpha_1) \leq \beta_\nu^{(t)}(\alpha_2). \tag{16}$$
3) Fix α, p_{maj}, and denote $\nu_1 = (p_{\text{xor},1}, p_{\text{maj}})$, $\nu_2 = (p_{\text{xor},2}, p_{\text{maj}})$. If the function $p_\oplus \to P_{e,\nu}(\alpha)$ is increasing with $p_\oplus$, then
$$\forall t > 0, \quad p_{\text{xor},1} \leq p_{\text{xor},2} \ \Rightarrow \ \beta_{\nu_1}^{(t)}(\alpha) \leq \beta_{\nu_2}^{(t)}(\alpha). \tag{17}$$
4) Fix α, $p_\oplus$, and denote $\nu_1 = (p_\oplus, p_{\text{maj},1})$, $\nu_2 = (p_\oplus, p_{\text{maj},2})$. If the function $p_{\text{maj}} \to P_{e,\nu}(\alpha)$ is increasing with p_{maj}, then
$$\forall t > 0, \quad p_{\text{maj},1} \leq p_{\text{maj},2} \ \Rightarrow \ \beta_{\nu_1}^{(t)}(\alpha) \leq \beta_{\nu_2}^{(t)}(\alpha). \tag{18}$$

Proof:

1) The proof is made recursively.

First compute $\beta_\nu^{(2)}(\alpha) = \alpha(1 - 2P_{e,\nu}(\alpha)) + P_{e,\nu}(\alpha) \geq \alpha$. The inequality come from $(1 - 2P_{e,\nu}(\alpha)) > 0$. As $\beta_\nu^{(1)}(\alpha) = \alpha$, we get $\beta_\nu^{(2)}(\alpha) \geq \beta_\nu^{(1)}(\alpha)$.

Then assume that $\beta_\nu^{(t)}(\alpha) \geq \beta_\nu^{(t-1)}(\alpha)$ and compute

$$\begin{aligned}
&\beta_\nu^{(t+1)}(\alpha) - \beta_\nu^{(t)}(\alpha) \\
&= (1 - 2\alpha)\left(P_{e,\nu}\left(\beta_\nu^{(t)}(\alpha)\right) - P_{e,\nu}\left(\beta_\nu^{(t-1)}(\alpha)\right)\right) \\
&\geq 0.
\end{aligned}$$

The inequality come from the above assumption. At the end, we get $\beta_\nu^{(t+1)}(\alpha) \geq \beta_\nu^{(t)}(\alpha)$, which proves (15).

2) Compute

$$\begin{aligned}
&\beta_\nu^{(t+1)}(\alpha_2) - \beta_\nu^{(t+1)}(\alpha_1) \\
&= (1 - 2\alpha_1)P_{e,\nu}\left(\beta_\nu^{(t)}(\alpha_1)\right) + (1 - 2\alpha_2)P_{e,\nu}\left(\beta_\nu^{(t)}(\alpha_2)\right) \\
&\quad + (\alpha_1 - \alpha_2) \\
&\geq 0,
\end{aligned}$$

which proves (16)

3) The proof is made recursively. We first show that $P_{e,\nu}\left(\beta_{\nu_1}^{(1)}(\alpha)\right) \leq P_{e,\nu}\left(\beta_{\nu_2}^{(1)}(\alpha)\right)$ and that $\beta_{\nu_1}^{(2)}(\alpha_2) \leq \beta_{\nu_2}^{(2)}(\alpha)$. We then assume that at step t, $P_{e,\nu}\left(\beta_{\nu_1}^{(t)}(\alpha)\right) \leq P_{e,\nu}\left(\beta_{\nu_2}^{(t)}(\alpha)\right)$ and $\beta_{\nu_1}^{(t)}(\alpha_2) \leq \beta_{\nu_2}^{(t)}(\alpha)$, and we show that this is also true at step $t + 1$.

The proof for 4) is the same as the proof for 3).

■

Proposition 2 assumes that the function $P_{e,\nu}(\alpha)$ is increasing with α and with the decoder noise parameters. Although it is reasonable to assume that the error probability of the faulty decoder increases with the BSC parameter and with the decoder noise parameters,the results of [9], [10] show that the second assumption is not always true. For example, for the discrete Min-Sum decoder with 7 quantization levels for the messages, the authors of [9] observe that the noise in the decoder can sometimes improve the decoder performance compared to the noiseless case. The same effect is observed for the Probabilistic Gradient Descent Bit-Flipping decoders introduced in [10]. As a consequence, Proposition 2 does not hold for such decoders. On the other hand, for the faulty OS-MAJ decoder, we can show that the function $P_{e,\nu}(\alpha)$ is increasing with α and with the decoder noise parameters. As a consequence, for the memory architecture we consider in the paper, higher values of α, $p_\oplus$, p_{maj}, will lead to increased degradation levels $\beta_\nu^{(t)}(\alpha)$.

Proposition 2 also shows that the successive degradation levels $\beta_\nu^{(t)}(\alpha)$ are increasing with t. As a result, even with the refresh decoder, the faulty memory keeps degrading the stored information. However, we hope that the sequence of degradation levels $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ converges to a fixed point which is no too high, so that the initial $\mathbf{x}^{(0)}$ can always be recovered from $\mathbf{x}^{(t)}$, even for large values of t. In order to verify this condition, we now analyze the convergence behavior of $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$.

D. Fixed-point analysis

Here, we analyze the asymptotic behavior of $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ by determining the fixed points of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$. The fixed points of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ are the values β satisfying $\beta = (1 - \alpha)P_{e,\nu}(\beta) + \alpha(1 - P_{e,\nu}(\beta))$, or equivalently if $\alpha \neq 1/2$,

$$P_{e,\nu}(\beta) = \frac{\beta - \alpha}{1 - 2\alpha}. \tag{19}$$

From the condition (19), the fixed points of the sequence of $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ correspond to the intersection of the curve representing $P_{e,\nu}(\beta)$ and the straight line $y = \frac{\beta-\alpha}{1-2\alpha}$. This gives a very simple condition to determine the fixed points of $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$. Note that if $P_{e,\nu}(1/2) = 1/2$ (which is always satisfied), then $\beta = 1/2$ is always a fixed point of $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$, whatever the value of α is.

The fixed points correspond to the possible limits of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$. We know that $\beta = 1/2$ is always a fixed point, but it is a bad one for which we cannot recover the original $\mathbf{x}^{(0)}$ from $\mathbf{x}^{(t)}$. Thus, we hope that the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ has other fixed points which correspond to degradation levels that can be handled by the final decoder. In the following, we propose a definition of the memory reliability that accounts for this condition.

IV. Reliability Conditions

In this section, we give a definition of the reliability of the memory. The reliability definition relies on the above asymptotic analysis on the sequence of degradation levels $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$. From the reliability conditions, we propose a threshold definition that determines the set of degradation parameters α that lead to a reliable storage of information.

A. Reliability Conditions

The reliability conditions we give here are based on the reliability conditions originally introduced in [3, Section 2.2]. The following definition adapts the reliability conditions of [3] to our analysis of the convergence properties of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$.

Definition 1: Consider the memory architecture of Section II and fix the parameters α, $p_\oplus$, p_{maj}. Consider the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ given in Proposition 1. Denote $\mathcal{B}$ the set of fixed-points of $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ excluding $1/2$, and denote by $\beta^\star$ the threshold of the noiseless Gallager B decoder.

A memory is said to be *reliable* for the parameters α, $p_\oplus$, p_{maj}, if the following three conditions are verified

1) *Bounded redundancy*: The redundancy R_{ed} of the memory does not depend on the memory capability k,
2) *Stability*: The set $\mathcal{B}$ is nonempty,
3) *Admissibility*: The set $\mathcal{B}$ is such that $\max \mathcal{B} \leq \beta^\star$.

The condition 1 requires bounded redundancy. From (7), The condition 1 is always fulfilled. The conditions 2 and 3 are

related to the asymptotic analysis of the successive degradation levels in the memory. The condition 2 requires that the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$ has fixed points other than $\beta = 1/2$. It guarantees that the memory is stable in the sense that the degradation levels converge to a fixed point. The condition 3 ensures that the fixed point corresponds to a degradation level that can be handled by the final decoder.

The validity of Conditions 2 and 3 depend on the value of the memory degradation parameter α, and on the decoder noise parameters $p_\oplus$ and p_{maj}. In order to identify the set of parameters α, $p_\oplus$, p_{maj}, that lead to a reliable memory, we introduce a threshold definition as follows.

B. Threshold Definition

The following threshold definitions were introduced for LDPC codes in channel coding. The noiseless threshold in [11] was defined as the maximum channel parameter α such that $P_{e,\nu}(\alpha) = 0$. This condition cannot be applied here because of the noise introduced by the faulty hardware, which prevents the decoder from reaching an error probability 0. This is why several other threshold definitions were introduced for noisy decoders: the useful threshold [8], the target-BER threshold [8], [12], and the functional threshold [13]. However, these threshold definitions cannot be used in our context, because they characterize the behavior of the faulty decoder alone. Here, we introduce a new threshold definition that takes into account the dynamic of the whole memory architecture.

Definition 2: Consider the memory architecture of Section II and fix the decoder noise parameters $p_\oplus$, p_{maj}. The *degradation threshold* is defined as

$$\overline{\alpha} = \arg\max_{\alpha}\{\text{The memory is reliable in the sense of Definition 1}\}. \tag{20}$$

The degradation threshold is defined as the maximum parameter α for which the memory is reliable. In the above definition, the decoder noise parameters $p_\oplus$ and p_{maj} are fixed, and the threshold is only on α. Indeed, α is the parameter of the memory, while $p_\oplus$ and p_{maj} are the parameters of the correction circuit, and we want to express the threshold in terms of reliability of the memory elements.

At the end, the degradation threshold enables to characterize the set of parameters α, $p_\oplus$, p_{maj}, that lead to a reliable memory. These parameters can be represented in the form of reliability regions, as described in the following.

C. Reliability Regions

In this section, we provide reliability regions as the set of parameters α, $p_\oplus$, p_{maj}, that lead to a reliable memory. We consider regular LDPC codes of VN degree $d_v = 3$ and CN degrees $d_c = 4$, $d_c = 5$, $d_c = 6$, respectively. For the faulty OS-MAJ decoder, we set $p_\oplus = 10^{-3}$, and $p_{\text{maj}} = 10^{-3}$. In order to verify the reliability of the memory for given parameters α, $p_\oplus$, p_{maj}, we need the expression $P_{e,\nu}(\alpha)$ of the error probability of the faulty OS-MAJ decoder. Thus we first discuss the curves representing $P_{e,\nu}(\alpha)$ as a function of α.

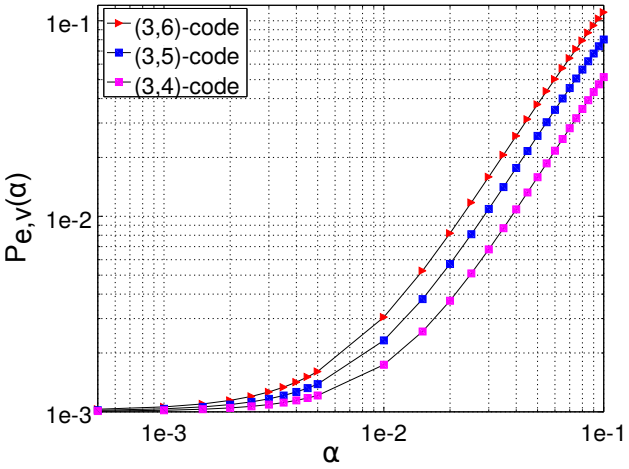

Fig. 2. Error probabilities w.r.t. α, for $p_\oplus = 10^{-3}$, $p_{\text{maj}} = 10^{-3}$ and Codes with $d_v = 3$

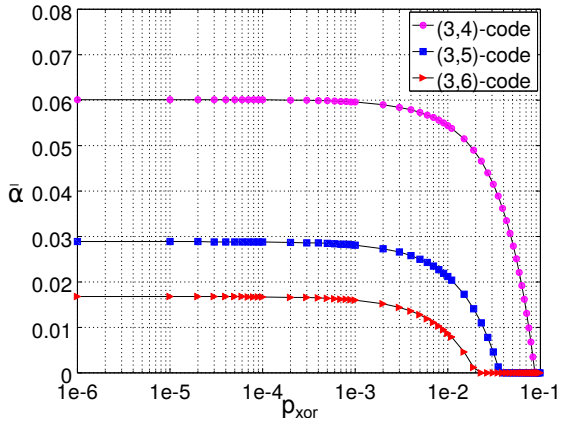

Fig. 3. Reliability regions w.r.t $p_\oplus$, for $p_{\text{maj}} = 10^{-3}$

The error probabilities $P_{e,\nu}(\alpha)$ are calculated from (11). Figure 2 shows $P_{e,\nu}(\alpha)$ as a function of α for the codes of VN degree $d_v = 3$. We see that the error probability increases with α but does it very slowly for small α. However, there is no distinguishable threshold value on α that would separate the low and the high error probability regions. However, from the analysis carried in the paper, we can identify a memory threshold by analyzing the properties of the sequence $\{\beta_\nu^{(t)}(\alpha)\}_{t=1}^{+\infty}$, as illustrated in the following.

For the codes with $d_v = 3$, Figure 3 represents the threshold values $\bar{\alpha}$ obtained from Definition 2 with respect to $p_\oplus$ Figure 3 thus gives the reliability regions with respect to $p_\oplus$. As expected, the reliability regions shrink with the code rate increase. The reliability regions are convex, even for large values of $p_\oplus$. When the decoder noise parameter $p_\oplus$ becomes too large, the threshold value $\bar{\alpha}$ becomes 0, which means that the decoder noise is too high to enable the memory to be reliable . Figure 4 represents the reliability regions with respect to p_{maj}. We get the same conclusions as before.

We conclude this section by a remark that the analysis of the paper and the threshold definition enable characterization

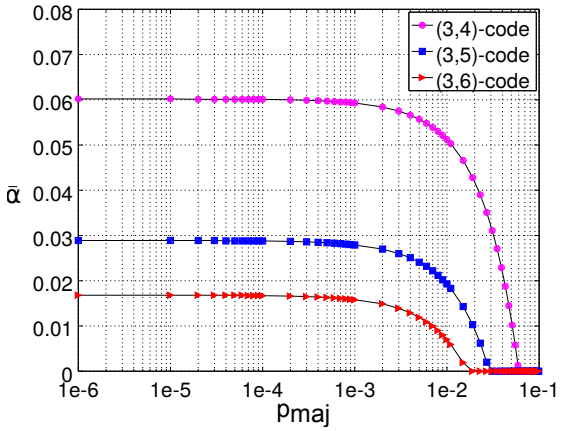

Fig. 4. Reliability regions w.r.t p_{maj}, for $p_{\oplus} = 10^{-3}$

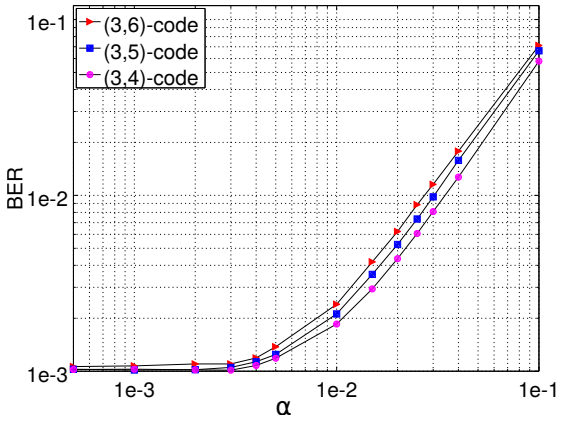

Fig. 5. BER w.r.t. α of the faulty OS-MAJ Decoder with $p_{\oplus} = 10^{-3}$ and $p_{\text{maj}} = 10^{-3}$

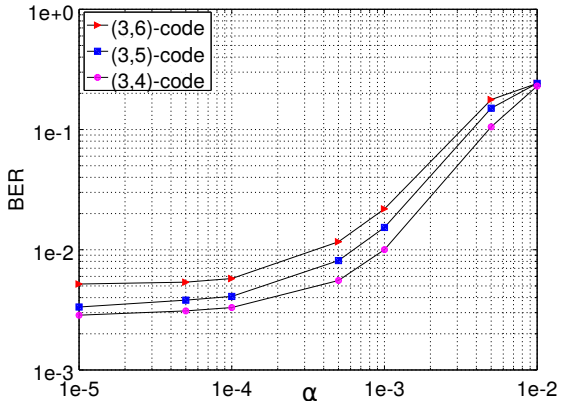

Fig. 6. BER w.r.t. α of the memory architecture with $p_{\oplus} = 10^{-3}$ and $p_{\text{maj}} = 10^{-3}$ and $T = 200$

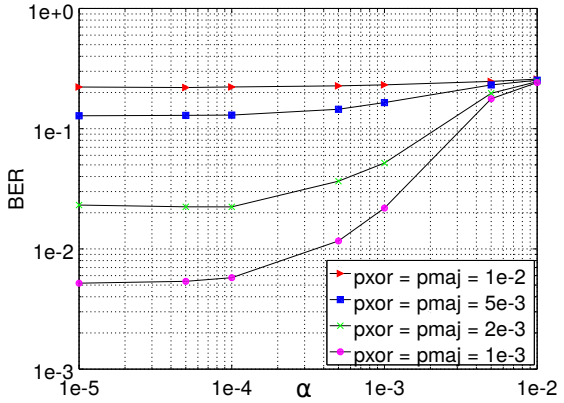

Fig. 7. BER w.r.t. α of the memory architecture with $p_{\oplus} = 10^{-3}$ and $p_{\text{maj}} = 10^{-3}$ and $T = 200$

of the set of parameters α, $p_{\oplus}$, p_{maj}, for which the memory is reliable. To verify the accuracy of the characterization, we now give finite-length simulation results.

V. Finite-length Simulations

In this section, we evaluate at finite length the reliability of the memory architecture we consider in the paper. We consider regular LDPC codes of dimension $k = 400$ with VN degree $d_v = 3$ and CN degrees $d_c = 4$, $d_c = 5$, $d_c = 6$, respectively.

Figure 5 represents the BER of the Faulty OS-MAJ Decoder for the three considered codes with $p_{\oplus} = 10^{-3}$ and $p_{\text{maj}} = 10^{-3}$. The curves are very similar to the curves of Figure 2 that represent the error probabilities with respect to α.

Figure 6 represents the BER for the whole memory architecture after $T = 200$ time instants for the three codes with $p_{\oplus} = 10^{-3}$ and $p_{\text{maj}} = 10^{-3}$. As expected, when the rate of the code increases, the BER increases. We see that when α becomes too large, the BER after $T = 200$ is too high and the memory is not reliable anymore. This effect comes for values of α that are smaller than the threshold value. As for the analysis of LDPC codes, the analysis carried in the paper is an asymptotic analysis, which explains the difference.

Figure 7 represents the BER for the whole memory architecture after $T = 200$ time instants for the code with $d_c = 6$, for several values of $p_{\oplus}$ and p_{maj}. As expected, the memory is less reliable when the decoder noise increases. In particular, for $p_{\oplus} = p_{\text{maj}} = 10^{-2}$, the memory is not reliable, whatever the value of α.

VI. Conclusion

In this paper, we provided an analysis of the reliability of the memory architecture proposed by Taylor [3] and Kuznetsov [4]. We expressed the successive error probabilities in the memory and we introduced a threshold definition to characterize the set of memory degradation parameters and decoder noise parameters that lead to a reliable memory. The results of the paper can be extended to irregular codes and to other refresh decoders, such as faulty Gallager A or B decoders with a small number of iterations.

Acknowlegement

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

REFERENCES

[1] M. Ivkovic, S. Chilappagari, and B. Vasic, "Construction of memory circuits using unreliable components based on low-density parity-check codes," in *Global Telecommunications Conference*, 2006, pp. 1–5.

[2] S. Chilappagari, M. Ivkovic, and B. Vasic, "Analysis of one step majority logic decoders constructed from faulty gates," in *IEEE International Symposium on Information Theory*, July 2006, pp. 469–473.

[3] M. Taylor, "Reliable information storage in memories designed from unreliable components," *Bell System Technical Journal*, vol. 47, pp. 2299–2337, Dec. 1968.

[4] A. Kuznetsov, "Information storage in a memory assembled from unreliable components," *Problems of Information Transmission*, vol. 9, pp. 254–264, 1973.

[5] S. Brkic, P. Ivanis, and B. Vasic, "Analysis of one-step majority logic decoding under correlated data-dependent gate failures," in *IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2014, pp. 2599–2603.

[6] B. Vasic and S. Chilappagari, "An information theoretical framework for analysis and design of nanoscale fault-tolerant memories based on low-density parity-check codes," *IEEE Transactions Circuits Systems I, Regular Papers*, vol. 54, no. 11, pp. 2438–2446, Nov. 2007.

[7] S. Chilappagari and B. Vasic, "Fault tolerant memories based on expander graphs," in *Information Theory Workshop*.

[8] L. Varshney, "Performance of LDPC codes under faulty iterative decoding," *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4427–4444, July 2011.

[9] C. K. Ngassa, V. Savin, and D. Declercq, "Unconventional behavior of the noisy min-sum decoder over the binary symmetric channel," in *Information Theory and Applications Workshop*, Feb. 2014, pp. 1–10.

[10] O. Rasheed, P. Ivanis, and B. Vasic, "Fault-tolerant probabilistic gradient-descent bit flipping decoders," *IEEE Communication Letters*, vol. 18, no. 9, pp. 1487 – 1490, September 2014.

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

[12] A. Balatsoukas-Stimming and A. Burg, "Density evolution for min-sum decoding of LDPC codes under unreliable message storage," *IEEE Communications Letters*, vol. 18, no. 5, pp. 849–852, May 2014.

[13] C. K. Ngassa, V. Savin, E.Dupraz, and D. Declercq, "Density Evolution and Functional Threshold for the Noisy Min-Sum Decoder," *Accepted at IEEE Transactions on Communications*, December 2014.