Error Errore Eicitur: A Stochastic Resonance Paradigm for Reliable Storage of Information on Unreliable Media

Predrag Ivaniš, *Senior Member, IEEE* and Bane Vasić, *Fellow, IEEE,*

***Abstract*—We give an architecture of a storage system consisting of a storage medium made of unreliable memory elements and an error correction circuit made of a combination of noisy and noiseless logic gates that is capable of retaining the stored information with lower probability of error than a storage system with a correction circuit made completely of noiseless logic gates. Our correction circuit is based on iterative decoding of low-density parity check codes, and uses the positive effect of errors in logic gates to correct errors in memory elements. In the spirit of Marcus Tullius Cicero's "Clavus clavo eicitur," ("one nail drives out another") the proposed storage system operates on the principle: "Error errore eicitur" - "one error drives out another." The randomness that is present in the logic gates makes these classes of decoders superior to their noiseless counterparts. Moreover, random perturbations do not require any additional computational resources as they are inherent to unreliable hardware itself. To utilize the benefits of logic gate failures, our correction circuit relies on two key novelties: a mixture of reliable and unreliable gates and decoder rewinding. We present a method based on absorbing Markov chains for the probability of error analysis, and explain how the randomness in the variable and check node update function helps a decoder to escape to local minima associated with trapping sets.**

***Keywords*—Faulty hardware, Gallager-B decoding, LDPC codes, Markov chains, unreliable logic.**

I. Introduction

Origins of a system's unreliability lie in the underlying physics mechanisms governing the operation of its parts. For example, in micro and nano-electronics devices it is due to low supply voltages and imperfections in the manufacturing process [1], [2], in space missions electronics due to high energy particles striking the semiconductor devices [3]. In order to ensure the robustness of a system to noise and/or faults in its parts, one relies on redundancy and computation, which compensate the negative effects of unreliable parts. Typical examples are systems for storage of information. Information written on a storage medium is physically represented as one of the several stable states of the memory elements comprising the medium (magnetization direction of magnetic grains, surface reflectivity, charge in capacitors, etc.). Since the reliability of the memory elements cannot be improved due to the underlying physics and manufacturing cost, one relies on a periodic refreshing of the stored data content to prevent the data decay. In this process, the errors that may have occurred in the medium are corrected in a computational device, called a correction circuit. Without loss of generality, a correction circuit can be assumed to perform a sequence of binary operations, and all traditional systems rely on the assumption that the correction circuit is noiseless, i.e., made of noiseless Boolean logic gates. In other words, computations performed in the correction circuit are deterministic, while randomness (in the form of noise and/or errors) exists only in the storage medium.

The above assumption is certainly appropriate for correction circuits in which the reliability of logic gates is many orders of magnitude higher than the reliability of memory elements. However, an interesting situation arises when a correction circuit itself is made of noisy components. For example, in low-powered submicron complementary metal–oxide–semiconductor (CMOS) chips mentioned above, the supply voltage is kept low in order to reduce power consumption, thus making logic gates susceptible to noise and increasing the probability of incorrect logic gate output. Due to unreliability of its logic gates, the correction circuit - whose purpose is to correct errors - introduces errors in the process of correcting errors from the storage medium.

Making logic gates reliable (for example by using larger supply voltages) appears as a logical solution. Another way to ensure robustness of a decoder is to employ the von Neumann multiplexing [4]. However, this comes with a price of large redundancy because it does not take into account the specifics of the decoding algorithm. The first attempt to use a more advanced coding scheme to ensure fault tolerance of storage systems made from unreliable components is due to Taylor [5] and Kuznetsov [6]. Fault-tolerant decoding and storage has attracted significant attention lately, and numerous approaches have been proposed which exploit the inherent redundancy of the existing decoders [7], [8], [9], [10]. In this paper we show that by two simple but key modifications – the rewinding schedule and more reliable gates for critical computations – the decoder can be made tolerant for a wide range of the gate failure rates. Moreover, we show that the logic gate errors may

This work was supported by the Seventh Framework Programme of the European Union, under Grant Agreement number 309129 (i-RISC project) and IUSSTF under the Indo-US Joint R&D Network Joint Centre on Data Storage Research/16-2014 Award. It is also funded in part by the NSF under grants CCF-0963726, CCF-1314147 and ECCS-1500170, and Serbian Ministry of Science under grant TR32028. Parts of this manuscripts were presented at the ITA Workshop 2016, and at the ISIT 2016.

B. Vasić is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, 85721 USA (e-mail: vasic@ece.arizona.edu), P. Ivaniš is with the School of Electrical Engineering, University of Belgrade, Serbia (e-mail:predrag.ivanis@etf.rs).

help the decoder, and present a class of noisy decoders that perform better than their noiseless counterparts.

We show that the correction circuit can operate in the unreliable gate regime, and still be able to correct more errors than the noiseless correction circuit. Moreover, the random perturbations do not require any additional computational resources as they are built in the unreliable hardware itself. User information is stored as a codeword of a low-density parity check (LDPC) code, and the correction circuit is based on the Gallager-B decoding algorithm (an architecture based on the bit-flipping will perform similarly [11], [12].) For the scheme to work, not all logic gates can be allowed to be noisy. Small fraction of critical ones are made reliable. This can be practically done using larger transistor size, higher voltage supply or slower clock. Secondly, to avoid accumulation of errors and divergence from the true codeword, the decoder is periodically rewound - i.e., re-initialized and restarted.

The first trace of the deliberate error idea can be found in Gallager's work where the random flips are used to resolve ties in the majority voting operation in the variable node, while the first iterative decoding algorithm that explicitly relies on randomness to correct errors is Miladinovic and Fossorier's Probabilistic Bit Flipping (PBF) [13]. A closely related technique of adding noise to messages in a BP decoder on the AWGN channel is by Leduc-Primeau *et al.* [14] for reducing error floor in the context of noiseless decoders. Recently it was shown by Sundararajan *et al.* [15] that random perturbations can be used to increase the performance of a gradient descent bit flipping decoder (GDBF), introduced by Wadayama *et al.* [16]. At the same time, we observed that the randomness coming from computational noise even more improves the GDBF decoding performance. Based on that result we developed a probabilistic gradient-descent bit flipping (PGDBF) algorithm [17], and introducing a random perturbation in the PGDBF algorithm is reminiscent of the operation of mutation in genetic algorithms [18].

A dominant method for analysis of noisy decoders is the noisy-density evolution (DE) technique [19]. It provides a convenient measure of robustness of a decoder, but it cannot provide an explanation why and when a noisy decoder performs better than the noiseless one. The underlying reasons are based on the assumptions of message independence and infinite number of iterations, thus implying an infinitely long code. For finite length codes, the performance of a noiseless iterative decoder depends upon the presence of trapping sets which are annihilated by the randomness in the decoder. Although the analysis in this paper is given for the storage system based on Gallager-B decoder, it is also applicable to other decoding algorithms.

The rest of the paper is organized as follows. In Section II, the preliminaries on iterative message passing decoding of LDPC codes are discussed. In Section III, we give a description of our storage system architecture. Section IV is dedicated to the theoretical analysis of the noisy Gallager-B decoder performance. The numerical results are presented in Section V. Finally, some concluding remarks and future research directions are given in Section VI.

II. Preliminaries

A. LDPC Codes

Consider a (γ, ρ)-regular binary LDPC code, denoted by (n, k), with code rate $R = k/n \geq 1 - \gamma/\rho$ and parity check matrix H. The parity check matrix is the bi-adjacency matrix of a bipartite (Tanner) graph $G = (V \cup C, E)$, where V represents the set of n variable nodes, C is the set of $n\gamma/\rho$ check nodes, and E is the set of $n\gamma$ edges. Each matrix element $H_{c,v} = 1$ indicates that there is an edge between nodes $c \in C$ and $v \in V$, which are referred as neighbors. Let $\mathcal{N}_v$ ($\mathcal{N}_c$) be the set of neighbors of the variable node v (check node c). Then, $|\mathcal{N}_v| = \gamma$, $\forall v \in V$ and $|\mathcal{N}_c| = \rho$, $\forall c \in C$, where $|\cdot|$ denotes cardinality. In irregular LDPC codes, nodes do not necessarily have the same number of neighbors.

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ denote a codeword of an LDPC code, where x_v represents the binary value associated with the variable node v. During transmission over a channel, an error vector $\mathbf{e} = (e_1, e_2, \ldots, e_n)$ is superimposed to the codeword, and the received word is $\mathbf{y} = \mathbf{x} \oplus \mathbf{e}$, where $\oplus$ is componentwise XOR, and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$. The analysis methodology presented here is not restricted to a particular type of channel errors nor error statistics, but the majority of the results are given for the Binary Symmetric Channel (BSC), where we assume that the code bits are flipped independently with probability α_M. The message passed from a check node c to a variable node v in the ℓ-th iteration is denoted by $\mu_{c\to v}^{(\ell)}$. The message passed from a variable node v to the check node c is denoted by $\nu_{v\to c}^{(\ell)}$.

Let $\mathbf{m}^{(\ell)} = \mu_{\mathcal{N}_v \setminus c \to v}^{(\ell)}$ denote all incoming messages to the variable node v except a message from the check node c. Similarly, $\mathbf{n}^{(\ell)} = \nu_{\mathcal{N}_c \setminus v \to c}^{(\ell)}$ denote all incoming messages to the check node c except from the variable node v.

B. Noiseless Gallager-B Decoder

The Gallager-B decoder works by sending binary messages over the edges of the graph. The messages are calculated based on the node update functions, following the rule that a message sent over an edge is obtained based on all received messages except the one arriving over that edge. The check node update function Ψ corresponds to the $(\rho - 1)$-input XOR logic gate, and $(\gamma - 1)$-input majority logic (MAJ) gate is used for the variable node update function implementation. In other words, the following operations are performed until the codeword is found or a maximum number of iteration, denoted by L, is reached. In each iteration ℓ, for each variable node $v \in V$, and for all $c \in \mathcal{N}_v$, i.e.

$$\nu_{v\to c}^{(\ell)} = \begin{cases} y_v & \ell = 0 \\ \Phi(y_v, \mathbf{m}^{(\ell-\mathbf{1})}) & \ell > 0 \end{cases}$$

where

$$\Phi(y_v, \mathbf{m}) = \begin{cases} \texttt{MAJ}(\mathbf{m}) & \texttt{MAJ}(\mathbf{m}) \neq 0 \\ y_v & \texttt{MAJ}(\mathbf{m}) = 0. \end{cases} \tag{1}$$

The function MAJ is defined as $\texttt{MAJ}(\mathbf{m}) = \text{sgn}(\sum \mathbf{m})$ wherein $\sum$ denotes the sum of its argument's components,

and sgn is the *signum* function. By convention, we take $\mathrm{sgn}(0) = 0$. We note that an alternative rule is possible which does not require a register for storing the channel values. In this case the previous variable estimate is used when there is a tie in the incoming messages of the variable node v. More precisely, $\nu_{v\to c}^{(\ell)} = \hat{x}_v^{(\ell-1)}$, when $\mathtt{MAJ}(\mathbf{m}^{(\ell)}) = 0$. Performance superiority of the rule given in Eq. (1), justifies the hardware overhead due to reliable registers for storing $\mathbf{y}$. Keeping the received word in a reliable temporary register is crucial for the decoder trajectory stabilization because it anchors the decoder state to that determined by the initial channel values, and prevents trajectory divergence from the initial value.

In each iteration $\ell \geq 0$, for each check node $c \in C$, and for all $v \in \mathcal{N}_c$, $\Psi(\mathbf{n}) = \prod \mathbf{n}$, where $\prod$ is taken componentwise and is equivalent to computing the XOR of the incoming bits. The decided bit value in the ℓ-th iteration is calculated by a majority vote among the all incoming messages together with the channel value. By convention, $+1$ corresponds to the input 0 and -1 to 1. For decoders with binary messages, it is more convenient to define the message alphabet as $\mathcal{M} = \{0, 1\}$. Then a check node c performs XOR operation, and it is said to be satisfied if $\sigma_c = 0$, and unsatisfied if $\sigma_c = 1$. The syndrome at the ℓ-th iteration is the ordered set $\{\hat{\sigma}_c^{(\ell)}\}_{c\in C}$ obtained from the codeword estimates, i.e., $\hat{\sigma}_c^{(\ell)} = \bigoplus \hat{\mathbf{x}}_{\mathcal{N}_c}^{(\ell)}$. A codeword is found if $\bigwedge_{c\in C} \hat{\sigma}_c^{(\ell)} = 0$, and the last operation requires an m-input AND gate. The output of the $\mathtt{MAJ}$ function is defined straightforwardly.

We say that a codeword is found if all M check nodes are satisfied. The iterative procedure is halted when all parity checks are satisfied or the predefined maximum number of iterations, denoted by L, is reached. The decoding is called successful if a codeword (either correct or wrong) is found. Otherwise, the decoding is said to have failed. The event of producing a codeword estimate which is a wrong codeword is called miss-correction. If t is the error correction capability of a given iterative decoder, the decoder is said not to converge to the correct codeword if there exists some error pattern $\mathbf{e}$ of weight $\mathrm{supp}(\mathbf{e}) > t$, which leads to a decoding trajectory $\{\mu_{C\to V}^{(\ell)}\}_{\ell\geq 0}$ such that for arbitrary ℓ there is at least one bit estimate $\hat{x}_v^{(\ell)}$ different from x_v. The support of a vector $\mathbf{x} = (x_1, x_2, ..., x_n)$, denoted by $\mathrm{supp}(\mathbf{x})$ is the set $\{x_i \mid x_i \neq 0\}$.

C. Noisy Gallager-B Decoder

Due to hardware unreliability the results of the update functions are not always correctly computed. We model this by XOR-ing the noiseless output with the binary error e_{MAJ} or $e_\oplus$

$$\begin{aligned} \nu_{v\to c}^{(\ell)} &= \Phi(y_v, \mathbf{m}^{(\ell)}) + e_{MAJ}^{(\ell)}, \\ \mu_{c\to v}^{(\ell)} &= \Psi(\mathbf{n}^{(\ell-1)}) + e_\oplus^{(\ell)}. \end{aligned} \tag{2}$$

We consider the von Neumann probabilistic failure model, according to which e_{MAJ} and $e_\oplus$ are independent Bernoulli random variables with parameters α_{MAJ} and $\alpha_\oplus$. Clearly, a gate with a smaller Bernoulli parameter are more reliable. Note that different realizations of Bernoulli random variables $e_{MAJ}^{(\ell)}$ and $e_\oplus^{(\ell)}$ correspond to different edges. To simplify the notation, the indices $v \to c$ and $c \to v$ are omitted.

In addition to the logic gates needed for calculation of messages, the decoder also requires logic gates for the final bit estimations and parity-checks calculation. If we allow the gates used to compute the variable node estimates to have failure rate comparable with the bit error rate of the decoder, the performance would be determined by the failure probabilities of these gates, not by the error control scheme [20]. Hence, these gates must have higher reliability.

III. The Proposed Storage System Architecture

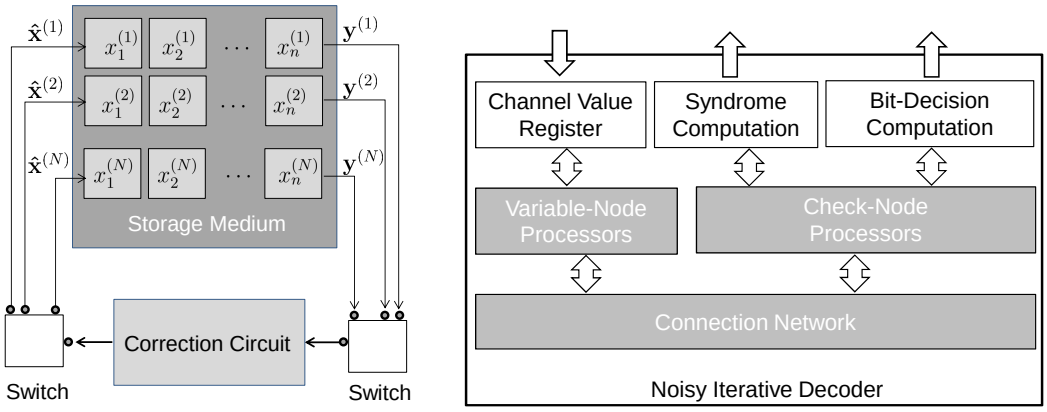
In our storage system, each k-bit user information is stored as a codeword of an (n, k) regular LDPC code of length n and code rate $R = k/n$ (applicability of our method to irregular codes is illustrated in Section V). The storage medium contains Nn memory elements and is capable of storing Nk user bits and $N(n-k)$ redundant bits. The memory elements are unreliable and fail transiently and independently of each other - they follow the von Neumann failure model [4]. To prevent data decay, the stored information is periodically refreshed. Each of N codewords is processed in a round-robin fashion by a common correction circuit, whose n-bit output is written back into the corresponding codeword location on the medium as shown in Fig. 1(a). The set $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\}$ in Fig. 1(a) denotes the set of codewords stored on a medium.

Due to errors in memory locations, $\mathbf{x}^{(i)}$ is read back as $\mathbf{y}^{(i)}$. The role of the correction circuit is to correct errors in each word $\mathbf{y}^{(i)}$, and write the n-bit codeword estimate $\hat{\mathbf{x}}^{(i)}$ back into the corresponding location on the medium. When a codeword on the medium is scheduled to be processed, the probability of error in each memory element is α_M. Since the correction of each codeword is independent of others, the storage of the codeword $\mathbf{x}$ can be modeled as a transmission through a communication channel. The medium is modelled as the BSC in which the code bits are flipped independently with probability α_M. The decoder must maintain a low probability of not recovering $\mathbf{x}$ - the so-called frame error rate (FER).

Critical Gates Must be Reliable: As explained at the end of Section II, the logic gates used to extract and output bits from any decoder must be more reliable than gates that are used for implementing the node update functions. Thus, the majority-logic gates in the decoder's decision logic are made of the more reliable gates, as explained at the end of the previous section. We also assume that the effect of errors in the encoder are incorporated in the value of the memory element error probability α_M.

The register inside the decoder which temporarily stores the word read from a memory medium (the channel values) is also reliable. This is necessary because otherwise the codeword estimate would drift away from the true codeword in the course of decoding, as iterations progress. As the channel value memory is usually much smaller than the memory required for the messages, the resulting overhead is quite small.

As syndrome checker (the ρ-input XOR gate together with the m-input AND gate) is used as a decoding halting criterion, it must be also made of reliable logic gates. Registers for



(a) (b)

Fig. 1: A block diagram of an information storage system with error correction coding. (a) A period-$1/N$ round-robin update of codewords where noisy Gallager-B decoder is applied as a correction circuit. (b) A block diagram of a noisy decoder (correction circuit). Shaded blocks are made of unreliable components, while white blocks are reliable.

storing the intermediate results of computations of Φ and Ψ are unreliable, and their unreliability is accounted for in α_{MAJ} and $\alpha_{\oplus}$. The blocks made on noiseless gates and those made on noisy gates are shown in Fig. 1(b).

Rewinding: To allow the decoder to benefit from errors, a large number of iterations are needed under some conditions of gate unreliability. However, too many logic gate errors can overwhelm the decoder. In addition to the Gallager-B update rules given in Eq. (1), our decoder is equipped with the following key feature which prevents the accumulation of errors in the messages when the number of iterations is large. If a codeword is not found after L_R iterations, where $L_R \ll L$, the decoding algorithm is re-initialized with the word received from the channel. Instead of running the whole L iterations, the decoder runs $r = L/L_R$ very short rounds with a maximum of L_R iterations each. A decoder with such rewinding schedule is referred to as the rewind-decoder and denoted by $\mathcal{F}^{\circlearrowleft^r}(L_R)$. We write this as $\mathcal{F}^{\circlearrowleft^r}(L_R) = \mathcal{F}^{\circlearrowleft^1}(L_R) \diamond \mathcal{F}^{\circlearrowleft^1}(L_R) \diamond \cdots \diamond \mathcal{F}^{\circlearrowleft^1}(L_R)$, where $\diamond$ denotes the rewinding schedule, and the expression has r terms. Clearly, the plain noisy Gallager-B decoder with no rewinding, denoted by $\mathcal{F}(L)$, is a special case of the rewinding decoder, $\mathcal{F}(L) = \mathcal{F}^{\circlearrowleft^1}(L)$.

IV. Performance Evaluation of Noisy Gallager-B Decoder

To characterize the FER performance of a given noisy decoder $\mathcal{F}$, we need to compute the probability $\Pr\{\hat{\mathbf{x}}^{(\ell)} \neq \mathbf{x}\}$ for arbitrary error vector $\mathbf{e}$ and find how it varies with the parameters $\alpha_{\oplus}$, α_{MAJ} and L, and, for the rewind-decoder $\mathcal{F}^{\circlearrowleft^r}$, also with the parameter of L_R. The goal is to find a region in this 4-dimensional design space in which $\mathcal{F}$ and $\mathcal{F}^{\circlearrowleft^r}$ outperform their noiseless counterpart $\overline{\mathcal{F}}$.

We now introduce a Markov model which facilitates this analysis. We first derive the model for the entire code graph, and then show that it can be simplified and applied to specific code subgraphs while keeping its accuracy. Let $\boldsymbol{\mu}^{(\ell)} = (\mu_c^{(\ell)})_{c \in C}$ be the ordered set of all messages from check nodes in iteration ℓ. From Eq. (2), they can be expressed as

$$\boldsymbol{\mu}^{(\ell)} = \Upsilon_C(\boldsymbol{\mu}^{(\ell-1)}) + \mathbf{e}_{\boldsymbol{\mu}}^{(\ell)}, \tag{3}$$

where the function Υ_C is the composition of Φ and Ψ, and define the dynamical system of the noiseless decoder. The binary vector $\mathbf{e}_{\boldsymbol{\mu}}^{(\ell)}$ of length $n\gamma$ $(= m\rho)$ is the realization of errors at time ℓ that affect the computation of messages $\boldsymbol{\mu}$. Their elements are deterministic functions of Bernoulli random variables representing the errors in the MAJ and XOR gates, and are time invariant but not independent. Probability distribution of $\mathbf{e}_{\boldsymbol{\mu}}$ can be determined using elementary probability.

From the discussion above, for a given decoder $\mathcal{F}$ and error pattern $\mathbf{e}$ in the memory elements, the random process $\{\boldsymbol{\mu}^{(\ell)}\}_{\ell > 0}$ is homogenous Markov chain $\mathcal{W}_{\mathbf{e}}$ with finite state space $S = \{0,1\}^{|C|\rho}$, the transition probability matrix $P_{\mathcal{W}} = (p_{\varepsilon,\delta})_{\varepsilon,\delta \in S}$ and stationary distribution $\pi_{\varepsilon}^{(\ell)}$.

Let $\pi_{\varepsilon}^{(\ell)} = \Pr\{\boldsymbol{\mu}^{(\ell+1)} = \varepsilon\}$ for some $\varepsilon \in S$, and let $\boldsymbol{\pi}^{(\ell)} = (\pi_{\varepsilon}^{(\ell)})_{\varepsilon \in S}$. The initial distribution $\boldsymbol{\pi}^{(0)}$ is a function of the memory error vector $\mathbf{e}$ and $\alpha_{\oplus}$. In the case of noiseless decoder, the initial state of the Markov chain $\bar{\varepsilon}_{\mathbf{e}}$ is uniquely determined for each channel error vector $\mathbf{e}$, and thus $\boldsymbol{\pi}_{\bar{\varepsilon}_{\mathbf{e}}}^{(0)} = 1$, while all other states have zero initial probability. In a noisy decoder, however, every outgoing message form a check node is also subject to a flip induced in that check node. Therefore, there is a nonzero probability of starting from any state in S. Given the channel error vector $\mathbf{e}$, the initial probability of the state ε is

$$\pi_{\varepsilon}^{(0)} = \alpha_{\oplus}^{d_{\bar{\varepsilon}_{\mathbf{e}},\varepsilon}} (1 - \alpha_{\oplus})^{|C|\rho - d_{\bar{\varepsilon}_{\mathbf{e}},\varepsilon}}.$$

where $d_{\bar{\varepsilon}_{\mathbf{e}},\varepsilon}$ is the Hamming distance between the binary vectors $\bar{\varepsilon}_{\mathbf{e}}$ and ε.

The transition probabilities between states $p_{\varepsilon,\delta} = \Pr\{\boldsymbol{\mu}^{(\ell)} = \delta | \boldsymbol{\mu}^{(\ell-1)} = \varepsilon\}$ depend on $\alpha_\oplus$ and α_{MAJ}. Since Φ is the function of the memory output $\mathbf{y}$, Υ_C also depends on $\mathbf{y}$. Thus the transition probabilities depend on the the channel error vector $\mathbf{e}$, and for a given decoder we have an ensemble of Markov chains $\{\mathcal{W}_\mathbf{e}\}_{\mathbf{e}\in\{0,1\}^n}$. Due to independence of logic gate errors, their impact can be condensed to a single parameter α_G (where G stands for "gate"). Let $\bar{\delta}_\mathbf{e} = \Upsilon_C(\varepsilon, \mathbf{e})$ be the state of the noiseless decoder $\overline{\mathcal{F}}$ reached from the state ε for the channel error vector $\mathbf{e}$. Then

$$p_{\varepsilon,\delta} = \alpha_G^{d_{\bar{\delta}_\mathbf{e},\delta}} (1-\alpha_G)^{|C|\rho - d_{\bar{\delta}_\mathbf{e},\delta}}, \tag{4}$$

where $d_{\bar{\delta}_\mathbf{e},\delta}$ is the Hamming distance between the binary vectors $\bar{\delta}_\mathbf{e}$ and δ, and α_G is the probability that a single XOR gate output in a noisy decoder $\mathcal{F}$ is different from the corresponding XOR gate output in the noiseless decoder $\overline{\mathcal{F}}$,

$$\alpha_G = \frac{1-\alpha_\oplus}{2}(1-(1-2\alpha_{MAJ})^{\rho-1}) + \frac{\alpha_\oplus}{2}(1+(1-2\alpha_{MAJ})^{\rho-1}).$$

The term multiplying $(1-\alpha_\oplus)$ is the occurrence probability of an odd number of MAJ gate errors, while the term multiplying $\alpha_\oplus$ is the occurrence probability of even number of MAJ gate errors. Note that for small logic gate error rates, the above expression can be approximated by

$$\alpha_G \approx (\rho-1)\alpha_{MAJ} + \alpha_\oplus.$$

Let the vectors $\hat{\mathbf{x}}^{(\ell)}$ and $\hat{\sigma}^{(\ell)}$ be the variable node decisions and check node estimates in iteration ℓ. If they were computed by noisy gates, the probability of convergence to a codeword would be close to zero for any iteration ℓ. This follows from Eq. (3) and the fact that $\boldsymbol{\pi}^{(\ell)} = \boldsymbol{\pi}^{(0)} P_{\mathcal{W}}{}^{(\ell)}$. This explains the requirement for using noiseless gates to compute the variable node estimates.

It is instructive to classify the states of $\mathcal{W}_\mathbf{e}$ with respect to their closeness to codewords. Due to channel, decoder, and the gate-error mechanism symmetries, the decoder behavior is independent of the received word $\mathbf{y}$. However, it depends on the error pattern $\mathbf{e}$ in memory elements. Furthermore, since the code is linear, it is sufficient to consider the all-zero codeword $\mathbf{x} = \mathbf{0}$ for the FER analysis [21]. Let $S_\mathbf{0} \subset S$ denote the subset of states for which all parity check are satisfied and the variable node decisions form the all-zero codeword $\mathbf{0}$. Similarly, $S_{\sim\mathbf{0}}$ denote the set of states for which all parity checks are satisfied, and the variable node decisions form a non-zero codeword. The set $S_{\sim\mathcal{C}}$ includes all states for which the variable node decisions are not codewords. Thus, the above three disjoint sets partition the set of states $S = S_\mathbf{0} \cup S_{\sim\mathbf{0}} \cup S_{\sim\mathcal{C}}$.

Note that the above analysis is applicable not only to the von Neumann model, but to any failure model in which gate failures are transient and happen with non-zero probability. In such a case the transition probabilities of the absorbing Markov chain would be non-zero, although the corresponding expressions could be more complex.

FER Performance and Time to Absorption for a Given Error Pattern: For a given error pattern $\mathbf{e}$, the conditional FER, and the conditional miscorrection rate (MER) of the decoder $\mathcal{F}$, in the iteration ℓ, $\Pr\{\hat{\mathbf{x}}^{(\ell)} \neq \mathbf{x}, \hat{\sigma}^{(\ell)} = \mathbf{0}\}$ can be now found from $\mathcal{W}$ and expressed as

$$\mathrm{FER}_\mathbf{e}^{(\ell)}(\mathcal{F}) = \Pr\{\boldsymbol{\mu}^{(\ell)} \in S_{\sim\mathbf{0}} \cup S_{\sim\mathcal{C}}\}$$
$$\mathrm{MER}_\mathbf{e}^{(\ell)}(\mathcal{F}) = \Pr\{\boldsymbol{\mu}^{(\ell)} \in S_{\sim\mathbf{0}}\}.$$

Theorem 1: For a noisy Gallager-B decoding algorithm $\mathcal{D} = \mathcal{F}(L)$ on any LDPC code $\mathcal{C}$, $\exists L^*$ and Δ, which depends on L^*, such that $\forall L > L^*$

$$|\mathrm{FER}_\mathbf{e}^{(L)}(\mathcal{D}) - \mathrm{MER}_\mathbf{e}^{(L)}(\mathcal{D})| < \Delta. \tag{5}$$

Proof: The proof is given in Appendix A. ∎

The average number of iterations taken for absorption to the states in $S_\mathbf{0}$ and $S_{\sim\mathbf{0}}$ can be calculated from the Markov chain $\mathcal{W}_\mathbf{e}$. If we combine all states in $S_\mathbf{0}$ into a single state, and do the same for $S_{\sim\mathbf{0}}$, we end up a with a reduced Markov chain denoted by $\mathcal{M}_\mathbf{e}$. The transition probability matrix of $\mathcal{M}_\mathbf{e}$

$$P = \begin{pmatrix} I_2 & \mathbf{0} \\ R & Q \end{pmatrix} \tag{6}$$

can be obtained from that of $\mathcal{W}_\mathbf{e}$ by summing up the corresponding rows as explained in Appendix A. $\mathcal{M}_\mathbf{e}$ is also homogenous and absorbing but has only two absorbing states, one corresponding to the correct decision, and the other corresponding to miscorrection. The matrix Q in Eq. (6) is a transition probability matrix between the transient states in $S_{\sim\mathcal{S}}$ with no zero entries, and I_2 is the 2×2 identity matrix. The transition probabilities from transient to absorbing states are given by the matrix $R = (R_\mathbf{0}, R_{\sim\mathbf{0}})$. The fundamental matrix $N = (I-Q)^{-1}$ determines the average times to absorption from different transient states. As long as the gate failures are transient, the sum of entries in any row of Q is strictly less then one, and the largest eigenvalue is less than one. If set of transient states is finite, the invertibility of $I-Q$ is guaranteed [22].

More specifically, $\sum_{\delta\in S_{\sim\mathcal{C}}} N_{\beta,\delta}$ is the average time to absorbtion from the transient state β. The complete derivation is provided in Appendix A. We note that the Markov chain was derived for a given channel error vectors $\mathbf{e}$, and that no assumptions were made about the statistics of $\mathbf{e}$. Hence, our methodology captures a wide set of memory models including memories with permanent errors (defects). In the rest of the paper, we focus our attention on the case when the bits in memory are flipped independently and transiently with probability α_M.

Average FER Performance: We have shown that the Markov chain model allows us to determine these probabilities. The details of this procedure are also given in Appendix. By averaging over all error patterns, we obtain the average FER as

$$\mathrm{FER}(\mathcal{D}) = \sum_{\mathbf{e}\in\{0,1\}^n} \Pr\{\mathbf{e}\} \times \mathrm{FER}_\mathbf{e}(\mathcal{D}). \tag{7}$$

Note that the Eq. (5) is valid for a given error pattern $\mathbf{e}$, and it translates to the averages (i.e., $\mathrm{FER}^{(L)} = \mathrm{MER}^{(L)}$) for

infinite L. However for finite L, the average $\mathrm{FER}^{(L)}(\mathcal{D})$ and $\mathrm{MER}^{(L)}(\mathcal{D})$ in Eq. (7) can differ significantly.

The probability of the error pattern $\mathbf{e}$, $\Pr\{\mathbf{e}\}$ depends on its Hamming weight $w(\mathbf{e})$. In the case of transient i.i.d. memory errors occurring with probability α_M, it can be expressed as

$$\Pr\{\mathbf{e}\} = \alpha_M^{w(\mathbf{e})}(1-\alpha_M)^{n-w(\mathbf{e})}. \tag{8}$$

For a noiseless decoder for which $\alpha_G = 0$, transitions between the states are deterministic, and the attractor basin of a dynamical system ($\boldsymbol{\mu}^{(\ell)} = \Upsilon_C(\boldsymbol{\mu}^{(\ell-1)})$ in Eq. (3) includes the codewords - which are the fixed points - and trapping sets, which can be either fixed points or cycle attractors. A noiseless decoder may oscillate between these states, thus failing to converge to a codeword. On the other hand, in a noisy decoder, every state can be reached with a nonzero probability. Thus, the noisy decoder will eventually converge to a codeword - either correct or incorrect one.

In the case when the decoding algorithm have small probability of miscorrection in the first decoding iterations, it is better to use the rewinding decoder with $r = L/L_R$ rounds, where the restarts and re-initializations are performed after L_R iterations, $L_R << L$. We denote this decoder by $\mathcal{D}^{\circlearrowleft} = \mathcal{F}^{\circlearrowleft^r}(L_R)$. The rewinding decoder is a composition of r rounds of the non-rewinding decoder $\mathcal{D} = \mathcal{F}(L_R)$ which runs for L_R iterations, i.e., $\mathcal{D}^{\circlearrowleft} = \mathcal{D}\diamond\mathcal{D}\diamond\cdots\diamond\mathcal{D}$. This fact allows us to obtain the miscorrection probabilities for every particular error pattern. The restart is performed only in the case of decoding failure, i.e. when the syndrome is not equal to zero. Therefore, the miscorrection probability of the rewinding decoder is

$$\mathrm{MER}_{\mathbf{e}}(\mathcal{D}^{\circlearrowleft}) = \sum_{k=1}^{r} \mathrm{MER}_{\mathbf{e}}(\mathcal{D})\left(\mathrm{FER}_{\mathbf{e}}(\mathcal{D}) - \mathrm{MER}_{\mathbf{e}}(\mathcal{D})\right)^{k-1}, \tag{9}$$

and the frame error rate after r rewinds includes the cases of miss-corrections in all r rounds and the case when the syndrome is not zero in all iterations. Therefore,

$$\mathrm{FER}_{\mathbf{e}}(\mathcal{D}^{\circlearrowleft}) = \mathrm{MER}_{\mathbf{e}}(\mathcal{D}^{\circlearrowleft}) + \left(\mathrm{FER}_{\mathbf{e}}(\mathcal{D}) - \mathrm{MER}_{\mathbf{e}}(\mathcal{D})\right)^{r}. \tag{10}$$

Finally, the average FER is obtained as

$$\mathrm{FER}(\mathcal{D}^{\circlearrowleft}) = \sum_{\mathbf{e}} \Pr\{\mathbf{e}\} \times \mathrm{FER}_{\mathbf{e}}(\mathcal{D}^{\circlearrowleft}). \tag{11}$$

We have shown that the ensemble of Markov chains $\{\mathcal{W}_{\mathbf{e}}\}$ completely determines the decoder's performance when the states are defined based on messages on *entire* Tanner graph. However, even for short codes, the state space S is large, and the above approach is computationally demanding. On the other hand, the theory of iterative decoders provides the graph topologies, known as trapping sets, that are responsible for decoding failures [23], [24]. As dominant trapping sets are typically the smallest ones, considering only messages on an *isolated* trapping set will result in a Markov chain with tractable number of states. Thus, instead of averaging $\mathrm{FER}_{\mathbf{e}}(\mathcal{D})$ over all error patterns (as in Eqs. (7), (11)), we estimate probabilities that the error pattern $\mathbf{e}$ in trapping set of critical classes χ is not corrected. By using expressions (1)-(3) from [25] and [26], the decoder performance in the error floor region is estimated as

$$\mathrm{FER}(\mathcal{D}) \geq \sum_{\chi} \sum_{w(\mathbf{e})=m}^{M} s|\chi| \frac{\binom{n}{w(\mathbf{e})}\binom{w(\mathbf{e})}{m}}{\binom{n}{m}} \Pr\{\mathbf{e}\}\, FER_{\mathbf{e}}^{(L)}(\mathcal{D}), \tag{12}$$

where w($\mathbf{e}$) is weight of error pattern, m and s denote the critical number and strength of the trapping set as defined in [25], M is the maximum number of errors which lead to decoding failure due to the trapping set χ, and $s\,|\chi|$ is number of non-isomorphic trapping sets of class χ [25].

V. Numerical Results

We start with an example of a short code for which the performance can be determined analytically, as it helps to explain the concepts we have introduced so far. We then present the results on the various column-weight three codes.

A. (5,1) Code – Analytical Approach

As a motivation example, we present performance analysis of a short irregular code defined with the bipartite graph presented in Fig. 2. It has the codeword length n=5, regular row weight ($\rho_j = 2$, $\forall j$), and irregular column weight. It is easy to check that there are only two codewords: "all-zero" codeword and "all-one" codeword. This is a repetition code with minimum Hamming weight $d_{min} = 5$, allowing the maximum likelihood (ML) decoder to correct all error patterns of weight $t = 2$ or less (this is a perfect code which attains the sphere packing bound).

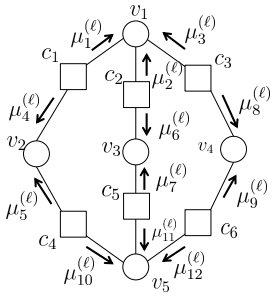
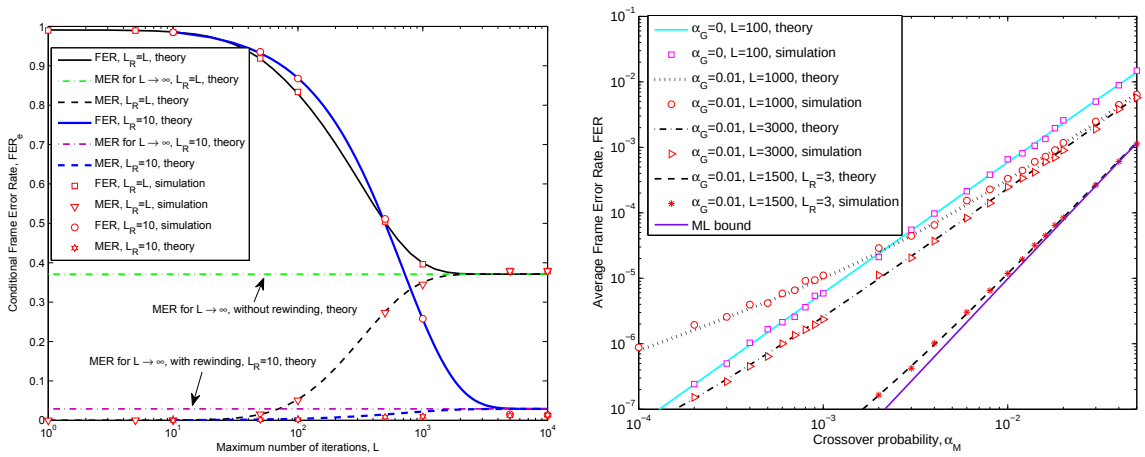


Fig. 2: Bipartite graph of repetition code (5,1).

As mentioned before, for the case of BSC and symmetric decoders, we can assume that the all-zero codeword is transmitted to analyze the effect of errors. First, we will consider weight-two error pattern $\mathbf{e}$=(11000). The initial state is determined by error pattern from the channel, and if messages $\mu_{c_j\to v_i}^{(1)}$ are sent from the j-th check node to the neighboring variable nodes, this initial state is $\boldsymbol{\mu}^{(0)} = (100, 10, 10, 10, 100)$. According to the messages $\mu_{c_j\to v_i}^{(1)}$, the codeword estimate after the first iteration is obtained as $\hat{\mathbf{x}}^{(1)} = (0, 1, 0, 0, 0)$. If all logic gates in the decoder are noiseless, the next state is completely determined by the check node and variable



(a) (b)

Fig. 3: (a) The conditional FER for error pattern $\mathbf{e}$=(11000) as a function of L for $\alpha_G = 0.01$ (with and without rewinding), (b) The average FER as a function of the crossover probability α_M for various values of α_G.

node processor functions. Therefore, the next state $\boldsymbol{\mu}^{(1)} = \bar{\delta}_{\mathbf{e}} = (000, 00, 00, 00, 111)$ is reached with probability one, i.e. $\Pr\{\boldsymbol{\mu}(\ell+1) = \bar{\delta}_{\mathbf{e}} | \boldsymbol{\mu}(\ell+1) = \varepsilon_{\mathbf{e}}\} = 1$, while the other transitions are forbidden. Syndrome is not equal to zero in further iterations, and this error pattern is not correctable by using noiseless decoder.

In the case of noisy Gallager-B decoder, the $(\rho_j - 1)$-input XOR gates which generate the messages $\mu^{(\ell)}_{c_j \to v_i}$ and $(\gamma_i - 1)$-input MAJ gates which generate messages $\nu^{(\ell)}_{v_i \to c_j}$ may be noisy. It is only assumed that the γ_i-input MAJ gates, which produce the codeword estimate are noiseless, as well as the syndrome checker logic. For such a noisy decoder, all other states $\boldsymbol{\mu}(\ell+1) = \delta_{\mathbf{e}}$ can be reached in the next iteration in addition to the state $\boldsymbol{\mu}(\ell+1) = \bar{\delta}_{\mathbf{e}}$. The transition probabilities $\Pr\{\boldsymbol{\mu}(\ell+1) = \delta_{\mathbf{e}} | \boldsymbol{\mu}(\ell+1) = \varepsilon_{\mathbf{e}}\}$ are given in Eq. (4). As we have shown in Section IV, by using a theory of absorbing Markov chains, it is possible to analytically determine the conditional FER and MER of a noisy decoder $\mathcal{F}(L)$ for any memory error vector $\mathbf{e}$.

Performance of noisy Gallager-B decoder for error pattern $\mathbf{e}$=(11000) is presented in Fig. 3(a). It is clear that the increase in the number of iterations results in better performance for the certain values of the failure rates. For an infinite number of decoding iterations, the probability that the pattern is not decoded converges to the corresponding MER, and if ℓ tends to infinity we obtain $\mathrm{FER}^{(\infty)}_{\mathbf{e}} = \mathrm{MER}^{(\infty)}_{\mathbf{e}}$. The accuracy of the proposed analytical model is verified for any number of iterations, by using independent simulation model.

In the same figure, we show that rewinding reduces the miscorrection probability. As MER lower bounds FER, the decoder performance can be improved for any error pattern if rewinding is applied with the appropriately chosen rewinding period L_R. If rewinding is applied in the moment where the probability of the miscorrection is negligible and the probability of correct decoding is not negligible, this will affect the overall performance after rewinding. As the numerical values for conditional FER and MER are known for a particular error pattern for all iterations, the optimal value of parameter L_R can be estimated. Numerical results obtained by using the proposed analytical approach perfectly corresponds to the simulation results.

The average FER as a function of crossover probability is presented in Fig. 3(b), and it is obtained by averaging over all error patterns i.e. by using Eq. (7). If the failure rate is high, it can help us to decode some high-weight patterns uncorrectable by using noiseless decoder but the MER for low-error patterns is increased, and some error patterns correctable by using noiseless decoder are now uncorrectable with high probability. Therefore, in the case of high failure rates, the performance in error-floor region is typically poor. For a lower failure rate, the average FER will be reduced for a wide range of α_M, with the price of increased number of iterations (effect from Fig. 3(a)). If the failure rate in logic gates is high, the decoder performance can be significantly improved by using the rewinding procedure. For the noisy decoding with $\alpha_{\oplus} = 0.01$, rewinding after $L_R = 3$ iterations results in performance close to the ML bound.

B. Annihilation of trapping sets by gate failures

For the received word $\mathbf{y}$, the subgraph induced by the set of variable nodes which are not eventually correct is called a trapping set. TS(a, b) denotes a trapping set with a variable nodes, and b odd-degree check nodes. TS$(5, 3)$ and TS$(4, 4)$ are shown in Fig. 4 (a) and Fig. 4 (b). ●/○ denotes an initially incorrect/correct variable node, and ■/□ denotes an even-degree unsatisfied/satisfied check node. ■ denotes an odd-degree check node.

The topology of TS$(5, 3)$ shown in Fig. 4(a) is similar to the graph in Fig. 2. The difference is in three additional

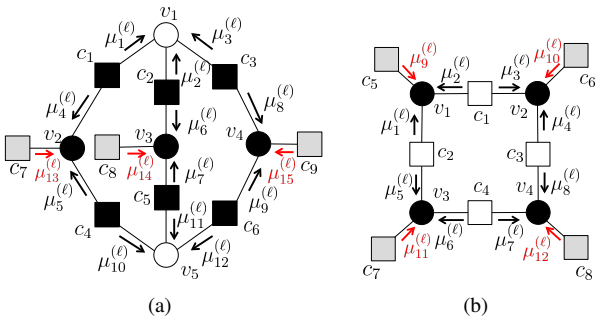


Fig. 4: Isolated trapping sets: (a) TS$(5,3)$, (b) TS$(4,4)$.

check nodes that connect this subgraph with the rest of the graph. Now, let us assume the following simplified scenario commonly used in literature to analyze trapping sets referred to as *independence assumption* [21]. In this scenario (i) messages generated inside the trapping set do not return from the rest of the graph, and (ii) gate failures that originate from the rest of the graph are represented by failures generated in the check nodes that connect the trapping set with the rest of the graph. Such check nodes are denoted by ■ in Fig. 4, and for the analysis of an isolated TS$(5,3)$ the Markov chain states are determined by $n\gamma = 15$ bits.

Using the above assumption, we now estimate the probability of "escaping" from TS$(5,3)$ that is induced by the three-bit error pattern given in Fig. 4(a). This is the only combination of three bits in the variable nodes $\{v_1, v_2, \ldots, v_5\}$ uncorrectable by the noiseless decoder. The corresponding initial state is $(\mu_1^{(0)}, \mu_2^{(0)}, ..., \mu_{15}^{(0)}) = (1,1,1,0,0,0,0,0,0,1,1,1,0,0,0)$. In order to exhibit a net performance gain, the noisy decoder must be able to correct more errors - both those correctable and those uncorrectable by the noiseless decoder. Therefore, we also analyze the effect of logic gate errors to error patterns correctable by a noiseless decoder. An example of such pattern is the one where the variables v_1, v_3 and v_5 are in error.

Fig. 5 shows the probability of correcting an error pattern on an isolated trapping set as a function of the number of iterations for different values of α_G, for TS$(5,3)$ and TS$(4,4)$. It can be seen that while larger α_G reduces the decoding latency for those error patterns uncorrectable by a noiseless decoder, the gate failures have a negative effect on decoding of error patterns correctable by the noiseless decoder.

The analysis in this section is given for isolated trapping sets and for the critical error patterns. If the trapping set is not isolated, logic gate failures in the rest of the graph could further degrade the overall performance. This effect is strong at very high failure rates when the gate failures introduce more new errors than what can be corrected within the trapping set. However, we show in the next section that the main conclusions remain the same even if the logic gate failures are inserted in the entire graph if the code have good distance properties. In such a case there is a broad range of failure rates where noisy decoder outperforms its noiseless counterpart.

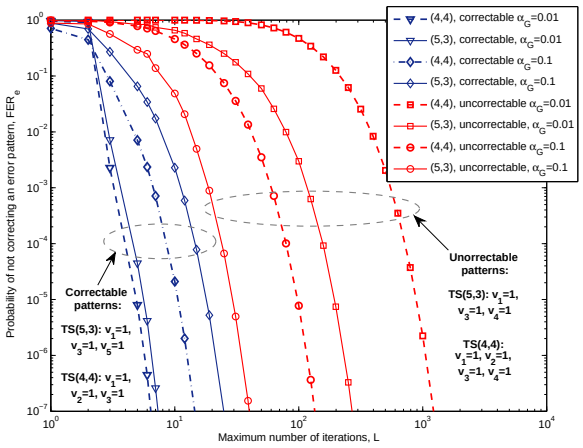


Fig. 5: Probability of not correcting an error pattern on isolated trapping sets TS$(5,3)$ and TS$(4,4)$, as a function of maximum number of iterations.

C. Simulation analysis of the finite length LDPC codes

In section V-B, we have shown that an isolated trapping set can be corrected by our noisy decoder with high probability. However, the isolation assumption means that the logic gate errors in the rest of the graph were ignored. Now, we show that the decoder performance improvement due to logic gate failures, which we observed and quantified on isolated subgraphs, also holds when logic gate failures are present in the entire Tanner graph.

We consider the regular $(n,k) = (155,64)$ LDPC code with $\gamma = 3$ and $\rho = 5$, constructed by Tanner [27]. The minimum Hamming distance between codewords is $d_{min} = 20$, thus the noiseless maximum likelihood (ML) decoder would be able to correct any nine-error pattern ($t = 9$). It was shown in [24] that TS$(5,3)$ is a dominant trapping set in the $(155,64)$ code, and the noiseless Gallager-B decoder fails on some three-error patterns [28]. The most harmful low-weight error pattern for this code is shown in Fig. 4(a). The conditional FER of the Tanner $(155,64)$ code for this pattern is presented in Fig. 6.

The particular low-weight error pattern, which cannot be decoded by the noiseless decoder, can be decoded with non-zero probability by a noisy decoder for a wide range of gate error probabilities. For high values of α_G, failures in logic gates make more damage than benefits as some error patterns correctable by the noiseless decoder need more iterations to be successfully decoded. Increasing the maximum number of iterations, L, reduces the probability that the error pattern remains uncorrected. The impact of L is more significant for high reliability gates. In this case, hardware errors cannot help much in the process of annihilating trapping sets because the state transition probabilities in $\mathcal{W}$ are small for most transitions other than those that already exist in the noiseless decoder. Consequently, the convergence to the subset S_0 takes longer, and the probability of escaping from a trapping set becomes significant only after a large number of iterations.

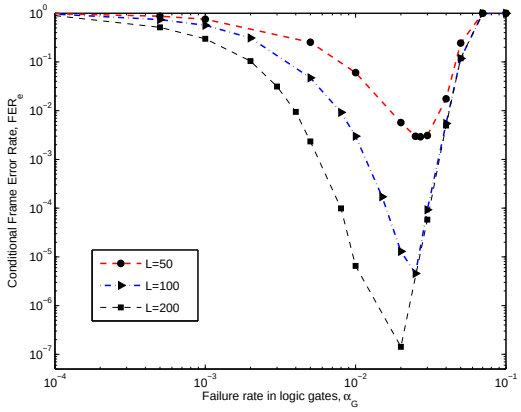
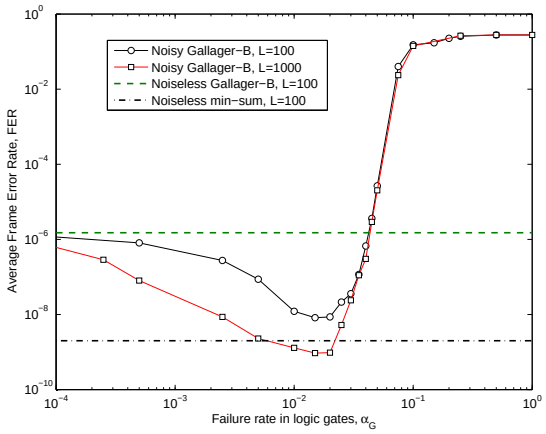


Fig. 6: Conditional FER as a function of failure rates for Tanner code (155,64), channel introduces a three errors-pattern which induces TS$(5,3)$.

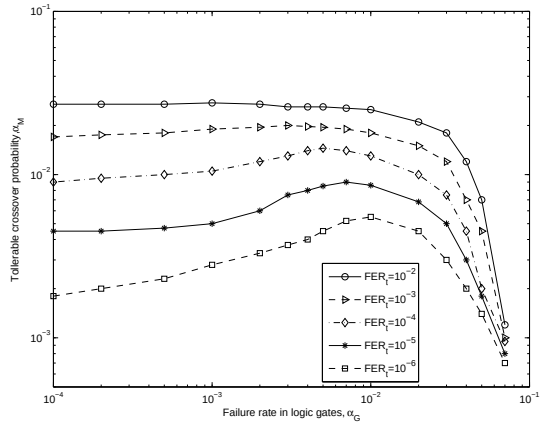
For this particular error pattern, the optimal value of failure rate is $0.01 < \alpha_G < 0.1$, which agrees with the results on an isolated trapping set shown in Fig. 5. The results in Fig. 6 are given for the error pattern that is uncorrectable by the noiseless decoder. Therefore, the noisy decoder works better on the TS$(5,3)$ than the noiseless one for any gate failure rate in the range $\alpha_G \leq 0.1$ and for any L. For a broad range of gate error rates, our decoder actually benefits from logic gate errors, and exhibits the stochastic resonance phenomenon.

If the storage medium is modeled as a BSC, any error pattern can occur at the decoder input with certain probability. We estimate the performance of noisy Gallager-B decoder in this case as well, for $\alpha_M = 2 \times 10^{-3}$. The numerical results are presented in Fig. 7(a) for the case when XOR and MAJ gates are both noisy. Since the previously analyzed error pattern is the most critical, the main effects are same as in Fig. 6, and the lowest FER is achieved for the failure rates that maximize successful correction of the most critical trapping sets. The noisy Gallager-B decoder is more efficient than its noiseless counterpart for any values of the failure rates in the logic gates less than $\alpha_G = 5 \times 10^{-2}$. Even more importantly, when $L = 1000$ and the gate error rates have near-optimal values, the noisy hard-decision algorithm has better performance than the more complex soft-decision min-sum algorithm realized in noiseless hardware. In the error floor region, L has the dominant effect on the FER, as shown in Fig. 7(a).

For a fixed hardware error rate α_G, we can identify a range of tolerable channel error rates α_M, required for not exceeding the predefined average FER after L decoding iterations. The numerical results are presented in Fig. 7(b). In the waterfall region, as expected, an increase of the logic gate failure rate *always reduces* the tolerable α_M. A similar effect was observed in [9]. However, the analysis in [9] was performed by using the density evolution technique. Since the density evolution is valid only for cycle-free Tanner graphs, it does not capture the stochastic resonance effect. Our analysis takes into account the effect of cycles through the analysis of trapping sets in the presence of logic gate failures. It identifies a range of α_G in which the logic gate failures *increase* the tolerance to channel errors in the error floor region.



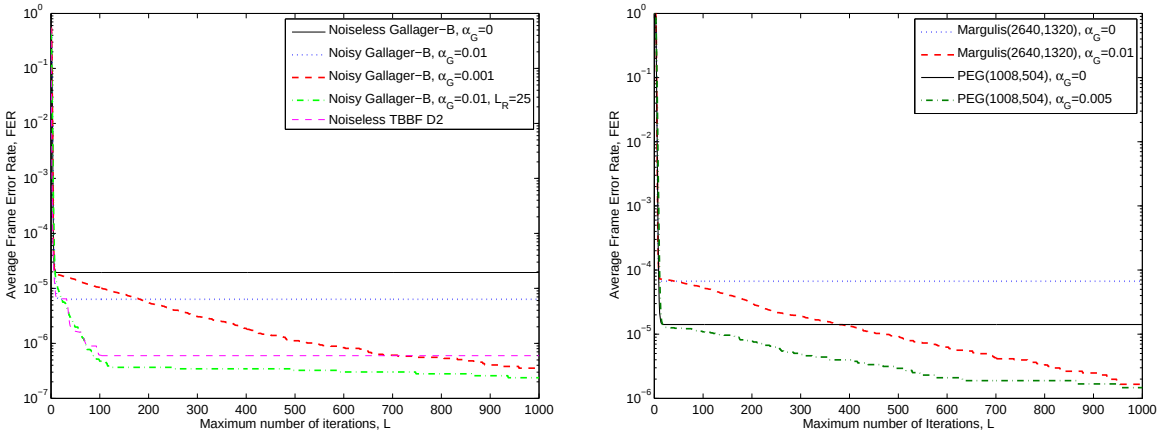
(a)



(b)

Fig. 7: Performance of the noisy Gallager-B decoder for Tanner $(155,64)$ code: (a) the average FER as a function of failure rates, storage medium is modeled as BSC with $\alpha_M = 2 \times 10^{-3}$, (b) the tolerable α_M in storage medium as a function of failure rates, for the fixed predefined target FER.

In Fig. 8(a), the average FER as a function of L is presented for $(155,64)$ code. The failure rate of $\alpha_G = 0.01$ results in the performance improvement after $L = 10$ iterations, while after $L = 30$ iterations, the performance becomes significantly better. However, prolonging the decoding further does not reduce the FER significantly. For smaller failure rates, significant performance improvement can be achieved only for large L, as shown in Fig. 5. The rewinding amplifies the positive effect of gate failures as it uses different initialization in every round. It also minimizes the negative effects of gate failures because it prevents accumulation of errors in large number of iterations, thus resulting in a faster performance improvement. It is interesting to notice that only $L = 100$ iterations are required for convergence when a noisy decoder with the failure rate $\alpha_G = 0.01$ is run in a rewinding schedule. This is slightly faster convergence than the convergence of the two-bit-bit-flipping (TBBF) decoding algorithm [29], that was optimized for the column weight three codes with girth $g = 8$.



(a) (b)

Fig. 8: The FER performance of the noisy Gallager-B decoder as a function of number of iterations (with and without rewinding) on the (a) Tanner $(155, 64)$ code for $\alpha_M = 5 \times 10^{-3}$, (b) Margulis $(2640, 1320)$ and PEG $(1008, 504)$ code for $\alpha_M = 1.5 \times 10^{-2}$.

In Fig. 8(b) we present the performance of two longer codes, Margulis $(2640, 1320)$ code and PEG $(1008, 504)$ code, both $(3, 6)$-regular LDPC codes. Although the dominant trapping sets are not the same for these codes [30], for both codes the performance improvement can be observed if $L \geq 20$. Allowing larger number of iterations results in additional performance improvement.

A comparison of different decoding strategies suitable for logic gates with high or low reliability is shown in Fig. 9. The FER curves for Tanner (155,64) code are shown in Fig. 9(a). In the error floor region, the numerical results obtained by using isolated trapping set analysis (by using Markov chains) combined with Eq. (12) perfectly match with the simulation results. For all L, the decoder $\mathcal{F}$ outperforms the ideal decoder $\overline{\mathcal{F}}$, and for large L, its performance approaches the $\lfloor \frac{d_{min}-1}{2} \rfloor$ bound. The positive effect of the rewinding is also shown for various values of $L = r \times L_R$ and various maximal number of iterations L.

Recently, Varshney have identified the fundamental limits for the construction of reliable memories by using noisy binary logic gates affected with i.i.d. errors, [31], and the importance of sphere packing bound is identified. In another recent paper [12], we have shown that the rewinding can be successfully applied to the various hard-decision decoders built of noisy gates, and that the maximum-likelihood bound could be approached if we can allow very large decoding latency.

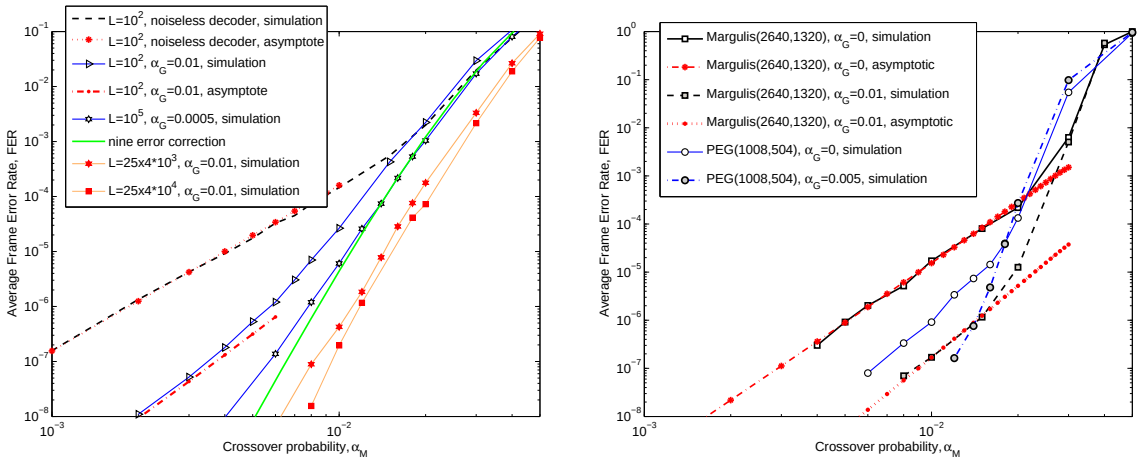
The FER curves of the PEG $(1008, 504)$ code and Margulis $(2640, 1320)$ code are shown in Fig. 9(b). For both codes, the error floor performance is greatly improved. Comparison with the analytical results obtained by using Eq. (12) is given for Margulis code. The PEG $(1008, 504)$ code suffers from a slight degradation in the waterfall region, but failures in logic gates result in significant performance improvement in error floor region. Although the numerical results are presented for only two values of α_G, the performance of both codes is improved for a wide range of logic gate failures rates.

VI. Conclusions and Perspectives

The decoder proposed in this paper is built of a mixture of noisy and noiseless logic gates, and for a broad range of gate failure rates it works better than a decoder made completely of noiseless gates. The fact that noise can be used constructively has been observed in many of natural and artificial analog signal processing systems, and is known as stochastic resonance [32]. The phenomenon studied in this paper can be also interpreted using the language of stochastic resonance. However, due to huge complexity of our correction circuits, the available stochastic resonance analysis tools are not sufficient to characterize their improved robustness.

The proposed analytical model based on absorbing Markov chains is used to quantify contributions of dominant trapping sets to the FER. Its parameters can be populated from the parameters of the failure mechanism statistics, and is used to identify the impact of critical parameters to the performance and offer design tradeoffs. For example, small gate failure rates require more iterations, but result in better performance. In the case when the rewinding is applied, it has been shown that the convergence speed is comparable to the best existing deterministic algorithm of a comparable complexity designed to escape from trapping sets. The analysis is applicable to any memory error statistics, including permanent failures. Although the results are given for von Neumann model of logic gate failures, we showed that the absorbing Markov chain with fully connected set of transient states can be defined for any type of transition failures in logic gates.

We have shown that the main effects of errors in logic gates can be captured by the analysis of isolated trapping sets, and that this analysis is predictable of the behavior on the entire Tanner graph. Even though all the XOR and MAJ gates used in the message update functions are subject to failures, for a broad range of failure rates the noisy decoder outperforms its noiseless counterpart. We have verified this by simulation of codes of various lengths obtained by different constructions.



(a) (b)

Fig. 9: Performance of the noisy Gallager-B decoder for as a function of the BSC crossover probability α_M for various decoding strategies effective for low and high failure rate ranges: (a) Tanner $(155, 64)$ code for various values of L, (b) Margulis $(2640, 1320)$ code and PEG $(1008, 504)$ code, $L = 1000$.

APPENDIX A (PROOF OF THEOREM)

Classification of States: It is instructive to classify the states of $\mathcal{W}_{\mathbf{e}}$ with respect to their closeness to codewords. Let $\mathcal{C}$ denote the set of all 2^k codewords and let $\sim \mathcal{C}$ denote the set of $2^n - 2^k$ n-tuples that are not codewords. For a given codeword $\mathbf{x}$, let $\sim \mathbf{x}$ denote the set $\mathcal{C} \setminus \mathbf{x}$. For a given decoder $\mathcal{F}$ and the error pattern $\mathbf{e}$ in the memory elements, let S be the set states of Markov chain $\mathcal{W}_{\mathbf{e}}$, and let $S_{\mathbf{x}}$ denote the subset of S for which all parity check are satisfied, and the variable node decisions form the codeword $\mathbf{x}$. Similarly, $S_{\sim\mathbf{x}}$ denotes the set of states for which all parity checks are satisfied, and the variable node decisions form a codeword different from $\mathbf{x}$. $S_{\sim\mathcal{C}}$ denotes the set of states for which the variable node decisions are not codewords. For any $\mathbf{x} \in \mathcal{C}$, the above three disjoint sets partition the set of states S

$$S = S_{\mathbf{x}} \cup S_{\sim\mathbf{x}} \cup S_{\sim\mathcal{C}}, \tag{13}$$

When the noiseless syndrome checker is turned on, and if the Markov chain is in the state $\beta \in S_{\mathbf{x}} \cup S_{\sim\mathbf{x}}$, the decoding is terminated, and the Markov chain stays in β. Thus, the states in $S_{\mathbf{x}}$ and $S_{\sim\mathbf{x}}$ are absorbing (the state transition diagram is shown in Fig. 10(a)).

Probability of Absorbtion.: Define now the matrices $P_{\sim\mathcal{C},\mathbf{x}}$, $P_{\sim\mathcal{C},\sim\mathbf{x}}$ and $P_{\sim\mathcal{C},\sim\mathcal{C}}$ with dimensions, respectively, $|S_{\sim\mathcal{C}}| \times 1$, $|S_{\sim\mathcal{C}}| \times 1$ and $|S_{\sim\mathcal{C}}| \times |S_{\sim\mathcal{C}}|$, as follows:

$$P_{\sim\mathcal{C},\mathbf{x}} = \left(\sum_{\varsigma \in S_{\mathbf{x}}} \Pr\{\boldsymbol{\mu}^{(\ell)} = \varsigma | \boldsymbol{\mu}^{(\ell-1)} = \beta\} \right)_{\beta \in S_{\sim\mathcal{C}}}$$

$$P_{\sim\mathcal{C},\sim\mathbf{x}} = \left(\sum_{\delta \in S_{\sim\mathbf{x}}} \Pr\{\boldsymbol{\mu}^{(\ell)} = \delta | \boldsymbol{\mu}^{(\ell-1)} = \beta\} \right)_{\beta \in S_{\sim\mathcal{C}}}$$

$$P_{\sim\mathcal{C},\sim\mathcal{C}} = \left(\Pr\{\boldsymbol{\mu}^{(\ell)} = \varepsilon | \boldsymbol{\mu}^{(\ell-1)} = \beta\} \right)_{\beta \in S_{\sim\mathcal{C}}, \varepsilon \in S_{\sim\mathcal{C}}}.$$

Note that the above matrices do not depend on ℓ due to homogeneity. The matrix

$$P = \begin{pmatrix} 1 & 0 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \\ P_{\sim\mathcal{C},\mathbf{x}} & P_{\sim\mathcal{C},\sim\mathbf{x}} & P_{\sim\mathcal{C},\sim\mathcal{C}} \end{pmatrix} = \begin{pmatrix} I_2 & \mathbf{0} \\ R & Q \end{pmatrix} \tag{14}$$

defines the transition probability matrix of a new Markov chain $\mathcal{M}_{\mathbf{e}}$. (Fig. 10(b)). In $\mathcal{M}_{\mathbf{e}}$, all the states in $S_{\mathbf{x}}$ are lumped into a single state. With a moderate abuse of notation, this new state is labeled as $S_{\mathbf{x}}$. The second absorbing state (with lumped states from $S_{\sim\mathbf{x}}$) is labeled as $S_{\sim\mathbf{x}}$. The matrix Q in Eq. (14) is a transition probability matrix between the transient states in $S_{\sim\mathcal{C}}$, and I_2 is the 2×2 identity matrix.

The initial distribution of $\mathcal{M}_{\mathbf{e}}$ can be written as

$$\boldsymbol{\pi}^{(\ell)} = (\pi_{\mathbf{x}}^{(\ell)}, \pi_{\sim\mathbf{x}}^{(\ell)}, \boldsymbol{\pi}_{\sim C}^{(\ell)}), \tag{15}$$

where $\boldsymbol{\pi}_{\sim C}^{(\ell)} = (\pi_{\beta}^{(\ell)})_{\beta \in S_{\sim C}}$ is the vector of initial probabilities of transient states, and the probabilities $\pi_{\mathbf{x}}^{(\ell)}$, $\pi_{\sim\mathbf{x}}^{(\ell)}$ are obtained by summing up the initial probabilities of the corresponding states: $\pi_{\mathbf{x}}^{(\ell)} = \sum_{\beta \in S_{\mathbf{x}}} \pi_{\beta}$, and $\pi_{\sim\mathbf{x}}^{(\ell)} = \sum_{\beta \in S_{\sim\mathbf{x}}} \pi_{\beta}$.

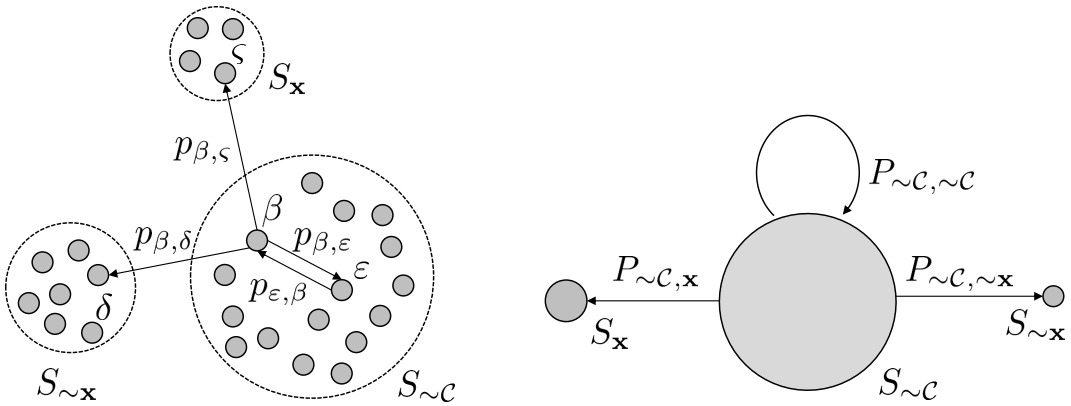
Note that the transition diagram of the transient states is a strongly connected graph, and that Q does not have any nonzero entries. The transition probabilities from transient to absorbing states $S_{\mathbf{x}}$ and $S_{\sim\mathbf{x}}$ are given by the matrix $R = (R_{\mathbf{x}}, R_{\sim\mathbf{x}})$, where $R_{\mathbf{x}} = P_{\sim\mathcal{C},\mathbf{x}}$, and $R_{\sim\mathbf{x}} = P_{\sim\mathcal{C},\sim\mathbf{x}}$.

The transition probabilities between states in ℓ iterations are given by

$$P^{\ell} = \begin{pmatrix} I_2 & \mathbf{0} \\ B^{(\ell)} & Q^{\ell} \end{pmatrix}, \tag{16}$$

where

$$B^{(\ell)} = \left(\sum_{i=0}^{\ell-1} Q^i \right) R = (I - Q)^{-1}(I - Q^{\ell+1})R. \tag{17}$$



(a) (b)

Fig. 10: Markov chain describing the iterative decoding process. (a) The state transition diagram of the Markov chain $\mathcal{W}_{\mathbf{e}}$ whose states are defined as ordered sets of messages from check nodes. (b) The reduced Markov chain, $\mathcal{M}_{\mathbf{e}}$. All the states from the original Markov chain that are in $S_{\mathbf{x}}$ are $S_{\sim\mathbf{x}}$ are lumped into a two corresponding absorbing states.

It is interesting to explore what happens for a very large number of iterations. Then

$$\lim_{\ell\to\infty} B^{(\ell)} = (I-Q)^{-1}R. \tag{18}$$

The fundamental matrix of the absorbing chain $N=(I-Q)^{-1}$ determines the average times to absorption from different transient states. More specifically, $\sum_{\delta\in S_{\sim\mathcal{C}}} N_{\beta,\delta}$ is the average time to absorption from the transient state β.

FER and MER: For a given decoder, and error pattern $\mathbf{e}$, the frame error probability and the miscorrection probability, in the iteration ℓ, $\Pr\{\hat{\mathbf{x}}^{(\ell)}\neq\mathbf{x}\}$ can be now found from $\mathcal{W}_{\mathbf{e}}$ and expressed as

$$\begin{aligned}\mathrm{FER}_{\mathbf{e}}^{(\ell)} &= \Pr\{\boldsymbol{\mu}^{(\ell)}\in S_{\sim\mathbf{0}}\cup S_{\sim\mathcal{C}}\}\\ \mathrm{MER}_{\mathbf{e}}^{(\ell)} &= \Pr\{\boldsymbol{\mu}^{(\ell)}\in S_{\sim\mathbf{0}}\},\end{aligned} \tag{19}$$

where $\mathbf{0}$ denotes the all-zero codeword.

If we write the matrix $B^{(\ell)}$ in Eq. (17) as $B^{(\ell)} = \left(B_{\mathbf{x}}^{(\ell)}, B_{\sim\mathbf{x}}^{(\ell)}\right)$, then it follows

$$\begin{aligned}\pi_{\mathbf{x}}^{(\ell)} &= \boldsymbol{\pi}_{\sim C}^{(0)} B_{\mathbf{x}}^{(\ell)}\\ \pi_{\sim\mathbf{x}}^{(\ell)} &= \boldsymbol{\pi}_{\sim C}^{(0)} B_{\sim\mathbf{x}}^{(\ell)}.\end{aligned}$$

From Eqs. (15) and (19), it follows

$$\begin{aligned}\mathrm{FER}^{(\ell)} &= 1-\pi_{\mathbf{x}}^{(\ell)}\\ \mathrm{MER}^{(\ell)} &= \pi_{\sim\mathbf{x}}^{(\ell)}.\end{aligned}$$

Because the sum of entries in every row of $\mathbf{Q}$ is strictly less than one, the largest eigenvalue of $\mathbf{Q}$ is less than one. Therefore, when ℓ tends to infinity $\mathbf{Q}^{\ell+1}\to 0$. Thus, $\boldsymbol{\pi}_{\sim C}^{(\ell)}=\mathbf{0}$, and thus $\pi_{\mathbf{x}}^{(\ell)}+\pi_{\sim\mathbf{x}}^{(\ell)}=1$, which leads to $\mathrm{FER}_{\mathbf{e}}^{(\infty)}=\mathrm{MER}_{\mathbf{e}}^{(\infty)}$.

REFERENCES

[1] S. Borkar, "Designing reliable systems from unreliable components: the challenges of transistor variability and degradation," *IEEE Micro*, vol. 25, no. 6, pp. 10–16, Nov. 2005.

[2] H. Jie and P. Jonker, "A system architecture solution for unreliable nanoelectronic devices," *IEEE Trans. Nanotech.*, vol. 1, no. 4, pp. 201–208, Dec. 2002.

[3] A. Campbell, P. McDonald, and K. Ray, "Single event upset rates in space," *IEEE Trans. Nucl. Sci.*, vol. 39, no. 6, pp. 1828–1835, Dec. 1992.

[4] J. V. Neumann, *Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components*, ser. Automata Studies. Princeton: Princeton University Press, 1956, pp. 43–98.

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

[6] A. V. Kuznetsov, "Information storage in a memory assembled from unreliable components," *Problems of Information Transmission*, vol. 9, no. 3, pp. 254–264, 1973.

[7] B. Vasić 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. Circ. Sys. I: Regular Papers*, vol. 54, no. 11, pp. 2438–2446, Nov. 2007.

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

[9] T. Yazdi, S. M. Sadegh, H. Cho, and L. Dolecek, "Gallager–B decoder on noisy hardware," *IEEE Trans. Commun.*, vol. 61, no. 5, pp. 1660–1673, May 2013.

[10] S. Brkic, P. Ivanis, and B. Vasić, "Analysis of one-step majority logic decoding under correlated data-dependent gate failures," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT 2014)*, Honolulu, Hawaii, June 29–July 4 2014, pp. 2599–2603.

[11] K. LeTrung, D. Declercq, and B. Vasić, "Analysis and efficient hardware implementation of probabilistic – GDBF," in *i-RISC Workshop: When Boole Meets Shannon*, Cork, Ireland, Sept. 1–2 2015, pp. 1–5.

[12] B. Vasić, P. Ivanis, D. Declercq, and K. LeTrung, "Approaching Maximum Likelihood Performance of LDPC Codes by Stochastic Resonance in Noisy Iterative Decoders," in *Proc. Inf. Th. and Applications Workshop 2016*, Feb. 2016.

[13] N. Miladinovic and M. Fossorier, "Improved bit-flipping decoding of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 4, pp. 1594–1606, April 2005.

[14] F. Leduc-Primeau, S. Hemati, S. Mannor, and W. Gross, "Dithered belief propagation decoding," *IEEE Trans. Commun.*, vol. 60, no. 8, pp. 2042–2047, Aug. 2012.

[15] G. Sundararajan, C. Winstead, and E. Boutillon, "Noisy gradient descent bit-flip decoding for LDPC codes," *IEEE Trans. Commun.*, vol. 62, no. 10, pp. 3385–3400, Oct. 2014.

[16] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, "Gradient descent bit flipping algorithms for decoding LDPC codes," *IEEE Trans. Commun.*, vol. 58, no. 6, pp. 1610–1614, June 2010.

[17] O.-A. Rasheed, P. Ivanis, and B. Vasić, "Fault-tolerant probabilistic gradient-descent bit flipping decoders," *IEEE Commun. Lett.*, vol. 18, no. 9, pp. 1487–1490, Sept. 2014.

[18] L. D. Davis and M. Mitchell, *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold, 1991.

[19] E. Dupraz, D. Declercq, B. Vasic, and V. Savin, "Finite alphabet iterative decoders robust to faulty hardware: Analysis and selection," in *8th Int. Symp. on Turbo Codes and Iterative Inform. Process. (ISTC)*, Bremen, Germany, Aug. 2014, pp. 107–111.

[20] B. Vasić, S. K. Chilappagari, S. Sankaranarayanan, and R. Radhakrishnan, "Failures of the Gallager B decoder: analysis and applications," in *Proc. 2nd Inf. Th. and Applications Workshop*. University of California at San Diego, Feb. 2006. [Online]. Available: http://ita.ucsd.edu/workshop/06/papers/160.pdf

[21] B. Vasić, D. Nguyen, and S. K. Chilappagari, "Chapter 6 - failures and error floors of iterative decoders," in *Channel Coding: Theory, Algorithms, and Applications: Academic Press Library in Mobile and Wireless Communications*. Oxford: Academic Press, 2014, pp. 299–341.

[22] S. I. Resnick, *Adventures in Stochastic Processes*. Basel, Switzerland, Switzerland: Birkhauser Verlag, 1992.

[23] T. J. Richardson, "Error floors of LDPC codes," in *Proc. 41st Ann. Allerton Conf. Commun., Contr. and Comp.*, Monticello, IL, USA, Sept. 2003, pp. 1426–1435.

[24] B. Vasić, S. Chilappagari, D. Nguyen, and S. Planjery, "Trapping set ontology," in *Proc. 47th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, USA, Sep. 30–Oct. 2 2009, pp. 1–7.

[25] S. K. Chilappagari, S. Sankaranarayanan, and B. Vasić, "Error floors of LDPC codes on the binary symmetric channel," in *Proc. IEEE International Conference on Communications (ICC '06)*, vol. 3, Istanbul, Turkey, 2006, pp. 1089–1094.

[26] S. Chilappagari, M. Chertkov, M. Stepanov, and B. Vasić, "Instanton-based techniques for analysis and reduction of error floors of LDPC codes," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 6, pp. 855–865, August 2009.

[27] R. M. Tanner, D. Sridhara, and T. Fuja, "A class of group-structured LDPC codes," in *Proc. ISTA*, 2001.

[28] S. Chilappagari and B. Vasić, "Error-correction capability of column-weight-three LDPC codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2055–2061, May 2009.

[29] D. V. Nguyen and B. Vasić, "Two-bit bit flipping algorithms for LDPC codes and collective error correction," *IEEE Trans. Commun.*, vol. 62, no. 4, pp. 1153–1163, Apr. 2014.

[30] M. Karimi and A. H. Banihashemi, "Efficient algorithm for finding dominant trapping sets of LDPC codes," *IEEE Trans. Inf. Theory*, vol. 58, no. 11, pp. 6942–6958, Dec. 2012.

[31] L. R. Varshney, "Toward limits of constructing reliable memories from unreliable components," in *Proc. IEEE Inf. Th. Workshop - Fall (ITW 2015)*, Oct. 2015, pp. 114–118.

[32] M. Kawaguchi, H. Mino, and D. Durand, "Stochastic resonance can enhance information transmission in neural networks," *IEEE Trans. Biomedic. Eng.*, vol. 58, no. 7, pp. 1950–1958, July 2011.