Gallager B LDPC Decoder with Transient and Permanent Errors

Chu-Hsiang Huang, Yao Li, and Lara Dolecek
Department of Electrical Engineering
University of California, Los Angeles
Los Angeles, USA
Email: {seanhuang0522, liyao}@ucla.edu, dolecek@ee.ucla.edu

***Abstract*—In this paper, the performance of a noisy Gallager B decoder used to decode regular LDPC codes is studied. We assume that the noisy decoder is subject to both transient processor errors and permanent memory errors. Due to the asymmetric nature of permanent errors, we model error propagation in the decoder via a suitable asymmetric channel. We then develop a density evolution type analysis on this asymmetric channel. The recursive expression for the bit error probability is derived as a function of the code parameters (node degrees), codeword weight, transmission error rate and the error rates of the permanent and the transient errors. Based on this analysis, we then derive the residual error of the Gallager B decoder for the regime where the transmission error rate and the processing error rates are small. In this regime, we further observe that the residual error can be well approximated by the sum of suitably combined transient errors and permanent errors, provided that the check node degree is large enough. Based on this insight we then propose and analyze a simple scheme for detecting permanent errors. The scheme exploits the parity check equations of the code itself and reuses the existing hardware to locate permanent errors in memory blocks. With high probability, the detection scheme discovers correct locations of permanent memory errors, while, with low probability, it mislabels the functional memory as being defective.**

I. Introduction

Prompted by the unavoidable defect-prone property of emerging nano-devices, understanding reliability of systems built out of unreliable components has become particularly important in the implementation of modern data processing algorithms. In this work, we jointly consider two major sources of hardware errors: (1) transient errors in combinatorial circuits made out of faulty gates, and (2) so-called 'stuck-at' (permanent) errors in memories holding intermediate data. Early work by Von Neumann [1] on transient errors opened up an active area of study on fault-tolerant computing, and has led to practical solutions based on additional (redundant) processing elements for combating transient errors [2], [3]. Taylor [4] investigated information storage capacities of memory built out of unreliable components. Heegard [5] studied a related problem of faulty hardware, namely that of faulty memories where certain components are permanently stuck in certain states. He also proposed a coding-theoretic technique to deal with stuck-at errors in memories. Further results on such coding methods were developed in [6] by Lastras-Montaño, Jagmohan, and Franceschini.

Recent exciting works investigated the performance of low-density parity-check (LDPC) decoders implemented on noisy hardware. The capacity and certain concentration results for a noisy LDPC message passing decoder were computed in [7] by Varshney. Message repetition was explored by Leduc-Primeau and Gross [8] to mitigate computational errors arising in a noisy Gallager B decoder. Density evolution of a sum-product message passing LDPC decoder is analyzed in [9] by Tarighati, Farhadi, and Lahouti. Our previous work on the fundamental performance limits of regular and irregular LDPC codes with binary and non-binary decoders implemented on noisy hardware with transient errors was reported in [10], [11].

The focus of this work is on a noisy Gallager B LDPC decoder where both transient errors in faulty gates and stuck-at errors in memory cells may occur. We refer to former as transient errors and to latter as permanent errors. This work extends the results of [10], [11] where only transient errors were considered. We first note that the nature of transient errors and permanent errors is quite different from each other. First, transient errors occur in computational units, whereas permanent errors occur in some specific (but unknown) memory cells. Under a transient error, the computational unit provides an output that is erroneous with some probability less than one, independent of all other computations. Under a permanent error, the memory unit repeatedly provides the same erroneous output.

The contributions of this paper are to develop an "asymmetric" density evolution of a noisy decoder, and to propose a "self-error-detecting" scheme that identifies defective memory cells. In Section II, we introduce our noisy decoder model. In Section III, density evolution of a faulty decoder is developed as a function of the code parameters, channel error rate, and transient and permanent error rates. We note that, with asymmetric errors, here induced by the presence of permanent errors, density evolution also depends on the codeword weight. In Section IV, based on the residual bit error derived from the density evolution equations, we propose a permanent error detection scheme. Further, we take advantage of the LDPC code structure to detect permanent errors without having to introduce additional circuitry for error detection. Performance analysis and simulation results show that the proposed detection scheme successfully detects permanent errors and improves the noisy decoder performance. Section V delivers the conclusions.

II. Modeling Channel, Transient and Permanent Errors

We consider the performance of a Gallager B decoder [12] subject to both transient errors due to unreliable processing units at variable and check nodes as well as permanent errors in memory cells that store intermediate messages. Here, defect memory cells are permanently stuck at logical 0 or 1. We focus on the decoder performance of the (d_v, d_c)-regular LDPC codes. Our results can be readily extended to the irregular case.

We consider the following sources of errors, capturing both the errors in transmission and errors in decoder:

- **Transmission errors**: Codewords are communicated over a binary symmetric channel with the cross-over probability ϵ.
- **Transient errors**: Whenever a variable-to-check message is calculated, with a probability q_v, the processor output is flipped. Whenever a check-to-variable message is calculated, with a probability q_c, the processor output is flipped.
- **Permanent errors**: The stuck-at error rates of memory cells storing decoder input are γ_0 and γ_1 for stuck-at-0 and stuck-at-1 errors, respectively. Likewise, the stuck-at error rates of memory cells storing variable-to-check messages (check-to-variable messages) are α_0 and α_1 (β_0 and β_1) for stuck-at-0 and stuck-at-1 errors, respectively. Note that we allow all permanent error rates to be different.

In the next section, we characterize the performance of the faulty Gallager B decoder by establishing proper density evolution equations.

III. Analysis of a Noisy Gallager B Decoder with Memory Failures

Under the general set-up wherein the permanent error rates may depend on the stuck value (i.e., stuck-at-1 and stuck-at-0 errors occur with different probabilities), we first observe that the popular assumption that considers the transmission of the all-zeros codeword no longer holds.

Let w denote the relative Hamming weight (i.e., the fraction of 1's in the codeword) of a codeword as the codelength goes to infinity. We develop density evolution[1,2] for two types of messages. The first type of messages are those propagating in or out of variable nodes associated with a transmitted codeword bit 1. The second type of messages are those propagating in or out of variable nodes associated with a transmitted codeword bit 0. We refer to these messages as type-1 and type-0 messages, respectively.

For $x \in \{0, 1\}$, we use $p_x^{(\ell)}$ to denote the error probability of a type-x variable-to-check message, i.e., the probability the message does not equal x, at iteration ℓ. Before presenting Theorem 1, which states the density evolution equations, we introduce some shorthand notation.

For convenience, we let $\alpha = \alpha_0 + \alpha_1$, $\beta = \beta_0 + \beta_1$, and $\gamma = \gamma_0 + \gamma_1$.

Next, we define the following auxiliary quantities. For $w \in (0, 1)$, and $x \in \{0, 1\}$, let

$$\begin{aligned}\hat{\phi}_{Cx}(z_0, z_1) =& \frac{1}{2} - \frac{\left(1 - 2z_0 - 2w(z_1 - z_0)\right)^{d_c - 1}}{2(1 + (-1)^x(1 - 2w)^{d_c - 1})} \\ &- (-1)^x \frac{\left(1 - 2z_0 - 2w(1 - z_1 - z_0)\right)^{d_c - 1}}{2(1 + (-1)^x(1 - 2w)^{d_c - 1})}\end{aligned} \tag{1}$$

be a function of z_0 and z_1 for $z_0, z_1 \in [0, 1]$. Additionally, we define

$$\hat{\phi}_C(z) = \frac{1 - \left(1 - 2z\right)^{d_c - 1}}{2}, \tag{2}$$

as a function of z for $z \in [0, 1]$.

The function $\hat{\phi}_{Cx}(z_0, z_1)$ represents the probability of error of a type-x message that is the output of a (hypothetical) noise-free check node. Here we assume that the input to this check node is comprised of $d_c - 1$ messages such that the probability of error for a type-0 incoming message is z_0 and the probability of error for a type-1 incoming message is z_1.

For $w = 0$, with probability 1, every message is type-0. As a result, the expression in (1) no longer depends on x and simplifies to $\hat{\phi}_C(z)$. Likewise, for $w = 1$, with probability 1, every message is type-1, and the same simplification applies. In addition, when the probability of error of a type-1 message is equal to that of a type-0 message, the two types of messages are considered indistinguishable, and again the same simplification to $\hat{\phi}_C(z)$ applies.

Let us also define

$$b'_x(z) = \left\lceil \left(d_v - 1 + \frac{\log\left(\frac{1 - (1-\gamma)\epsilon - \gamma_{1-x}}{(1-\gamma)\epsilon + \gamma_{1-x}}\right)}{\log \frac{1-z}{z}} \right) / 2 \right\rceil,$$

and let $b_x(z) = \max\{\lceil d_v/2, b'_x(z)\rceil\}$. For $x \in \{0, 1\}$, we then define

$$\begin{aligned}\hat{\phi}_{Vx}(z) = &((1 - \gamma)\epsilon + \gamma_{1-x}) \sum_{k=0}^{b_x(z)-1} \binom{d_v - 1}{k} (1 - z)^k z^{d_v - 1 - k} \\ &+ (1 - (1 - \gamma)\epsilon - \gamma_{1-x}) \sum_{k=b_x(z)}^{d_v - 1} \binom{d_v - 1}{k} z^k (1 - z)^{d_v - 1 - k}\end{aligned} \tag{3}$$

as a function of $z \in [0, 1]$ (and of $b_x(z)$).

The function $\hat{\phi}_{Vx}(z)$ represents the error probability of a type-x message that is the output of a (hypothetical) noise-free variable node. Here we assume that the input to this variable node is comprised of: (I) the initial decoder input whose probability of error is $p_x^{(0)} = \gamma_{1-x} + (1 - \gamma)\epsilon$, and (II) $d_v - 1$ type-x messages whose probability of error is z each. The parameter $b_x(z)$ is the variable node voting threshold in our noisy Gallager B decoder, and is at least $\lceil d_v/2 \rceil$ to guarantee a unique voting outcome. The parameter $b'_x(z)$ is the voting threshold that minimizes the expression of $\hat{\phi}_{Vx}(z)$ for a given z. Following derivations similar to those in [10], it can be shown that $b_x(z) = \lceil \frac{d_v}{2} \rceil$ when

$$z < p_x^{(0)} \quad \text{for} \quad x \in \{0, 1\}. \tag{4}$$

Since in density evolution, z is set to be $p_x^{(\ell)}$, the condition (4) is generally satisfied in the last few iterations given that the decoder input error rates $p_0^{(0)}$ and $p_1^{(0)}$ are sufficiently small, and that the decoder circuit error parameters are sufficiently small compared to decoder input error rates.

[1]Concentration results can be proved with the same tools as those in the proofs in [13], and are omitted here due to space limits.

[2]Density evolution for asymmetric channels was treated in [14] by averaging the density over all codewords in the same codebook. In contrast, we develop the exact density evolution for a codeword of a given relative weight w. Additionally, we also allow asymmetry inside the decoder.

Theorem 1. *Consider a codeword of relative weight w belonging to a (d_v, d_c)-regular LDPC code. Assume that the transmission and decoding errors are as specified in Section II. Then, for $x \in \{0,1\}$, the initial (iteration $\ell = 0$) bit error rate of a type-x message is*

$$p_x^{(0)} = \gamma_{1-x} + (1-\gamma)\epsilon.$$

At iteration $\ell \geq 0$, we consider two cases.

(a) For $0 < w < 1$ and $x \in \{0,1\}$, the bit error rates evolve recursively as

$$\begin{aligned} p_x^{(\ell+1)} =& \alpha_{1-x} + (1-\alpha)(q_v + (1-2q_v) \\ &\cdot \hat{\phi}_{Vx}(\beta_{1-x} + (1-\beta)(q_c + (1-2q_c)\hat{\phi}_{Cx}(p_0^{(\ell)}, p_1^{(\ell)})))). \end{aligned} \tag{5}$$

The average bit error rate, averaged over all codeword bits, at iteration ℓ is $p^{(\ell)} = (1-w)p_0^{(\ell)} + wp_1^{(\ell)}$.

(b) For $w = 1$ and d_c even or for $w = 0$, the average bit error rate equals the bit error rate of the type-w messages, and the recursion simplifies to

$$\begin{aligned} p^{(\ell+1)} =& p_w^{(\ell+1)} = \alpha_{1-w} + (1-\alpha)(q_v + (1-2q_v) \\ &\cdot \hat{\phi}_{Vw}(\beta_{1-w} + (1-\beta)(q_c + (1-2q_c)\hat{\phi}_{C}(p_w^{(\ell)})))). \end{aligned} \tag{6}$$

Proof: In the following, we assume $x \in \{0,1\}$.

Initially, at iteration $\ell = 0$, a type-x decoder input message is erroneous, i.e., does not equal x, iff (I) the message is stuck at $1-x$; or (II) there is no stuck-at error but the corresponding channel output is erroneous. Therefore,

$$p_x^{(0)} = \gamma_{1-x} + (1-\gamma)\epsilon.$$

At iteration $\ell \geq 0$, observe that a hypothetical noise-free check-to-variable message is erroneous iff there is an odd number of erroneous messages among the $d_c - 1$ incoming variable-to-check messages. Meanwhile, due to the checksum constraint, a type-1 (type-0) check-to-variable message has to be produced from an odd (even) number of type-1 incoming messages. As the codelength $n \to \infty$, the pdf of the number of type-1 messages among the $d_c - 1$ incoming messages can be approximated by the Binomial(d_c-1, w) distribution where w is the relative codeword weight. Thus, the probability that a check node has an odd number of incoming type-1 messages is

$$\sum_{\substack{0 \leq i \leq d_c-1 \\ i \text{ is odd}}} \binom{d_c-1}{i} w^i (1-w)^{d_c-i-1} = \frac{1}{2}(1-(1-2w)^{d_c-1}).$$

Meanwhile, the probability that an odd number of input messages are erroneous given that i of them are of type-1 and that the remaining $d_c - 1 - i$ of them are of type-0 is

$$\frac{1}{2}\big(1-(1-2p_1^{(\ell)})^i(1-2p_0^{(\ell)})^{d_c-i-1}\big).$$

Then, for $0 < w < 1$, by the law of total probability, the error probability of a hypothetical noise-free type-1 check-to-variable message at iteration ℓ, is then

$$\begin{aligned} \hat{q}_1^{(\ell)} =& \frac{1}{\frac{1}{2}(1-(1-2w)^{d_c-1})} \sum_{\substack{0 \leq i \leq d_c-1 \\ i \text{ is odd}}} \binom{d_c-1}{i} w^i(1-w)^{d_c-i-1} \\ &\cdot \frac{1}{2}\big(1-(1-2p_1^{(\ell)})^i(1-2p_0^{(\ell)})^{d_c-i-1}\big) \\ =& \hat{\phi}_{C1}(p_0^{(\ell)}, p_1^{(\ell)}). \end{aligned}$$

The expression for $\hat{\phi}_{C1}(p_0^{(\ell)}, p_1^{(\ell)})$ is provided in (1). The last equality is reached by applying the identity $\sum_{\substack{0 \leq i \leq k \\ i \text{ is odd}}} \binom{k}{i} a^i b^{k-i} = \frac{1}{2}((a+b)^k - (b-a)^k)$, setting $a = w(1-2p_1^{(\ell)})$, $b = (1-w)(1-2p_0^{(\ell)})$, and $k = d_c - 1$. Similarly we can derive $\hat{q}_0^{(\ell)} = \hat{\phi}_{C0}(p_0^{(\ell)}, p_1^{(\ell)})$, whose expression is also provided in (1).

Next, a type-x check-to-variable message is erroneous iff one of the following disjoint events occurs: (I) the message is stuck at $1-x$; or (II) there is no stuck-at error, but (IIa) the hypothetical noise-free message is correct but it is flipped due to a transient error, or (IIb) there is no transient error but the noise-free message itself is erroneous. The error probability of a noisy type-x check-to-variable message is then

$$q_x^{(\ell)} = \beta_{1-x} + (1-\beta)(q_c(1-\hat{q}_x^{(\ell)}) + (1-q_c)\hat{q}_x^{(\ell)}).$$

Then, given $q_x^{(\ell)}$, the error probability of a noise-free type-x variable-to-check message at the next iteration $l+1$, $\hat{p}_x^{(\ell+1)} = \hat{\phi}_{Vx}(q_x^{(\ell)})$, is derived following the same logic used in the proof of Theorem 1 of [10].

Finally, similar to the error rate derivation of check-to-variable messages as a function of the error rates of their hypothetical noise-free counterparts, we have

$$p_x^{(\ell+1)} = \alpha_{1-x} + (1-\alpha)(q_v(1-\hat{p}_x^{(\ell)}) + (1-q_v)\hat{p}_x^{(\ell)}). \tag{7}$$

By sequentially substituting $\hat{p}_x^{(\ell+1)}$, $q_x^{(\ell)}$, $\hat{q}_x^{(\ell)}$, and $p_x^{(0)}$ into (7), we arrive at (5).

For $w = 1$ and for $w = 0$, as noted previously, with probability 1, each message is of type-1 and type-0, respectively. Hence, density evolution is respectively on type-1 messages only and on type-0 messages only. Furthermore, for $w = 1$, if d_c is odd, with probability 1, the checksum constraints cannot be fulfilled, meaning that with probability 1 a codeword of relative weight w is not a valid codeword. Therefore, for $w = 1$, only the case where d_c is even is considered. Since the derivations of the $w = 1$ with even d_c case and the $w = 0$ case follow the procedures described above for the $w \in (0,1)$ case, the details are omitted.

■

While Theorem 1 provides the exact density evolution equations, when d_c is sufficiently large[3] and when all error rates specified in Section II are sufficiently small, the recursive equation (5) simplifies to

$$\begin{aligned} p_x^{(\ell+1)} \approx& \alpha_{1-x} + (1-\alpha)(q_v + (1-2q_v)\hat{\phi}_{Vx}(\beta_{1-x} \\ &+ (1-\beta)(q_c + (1-2q_c)\hat{\phi}_C((1-w)p_0^{(\ell)} + wp_1^{(\ell)})))), \end{aligned} \tag{8}$$

for $w \in (0,1)$ $x \in \{0,1\}$ and iteration $l \geq 0$.

To justify (8), we first show that for $w \in (0,1)$, and $(\frac{1}{2} - z_0)(\frac{1}{2} - z_1) > 0$, we have

$$\hat{\phi}_{C0}(z_0, z_1) \approx \hat{\phi}_{C1}(z_0, z_1) \approx \hat{\phi}_C((1-w)z_0 + wz_1). \tag{9}$$

From (1), we have

$$\begin{aligned} \hat{\phi}_{Cx}(z_0, z_1) =& \frac{1}{2} - \frac{\big(1-2z_0-2w(z_1-z_0)\big)^{d_c-1}}{2(1+(-1)^x(1-2w)^{d_c-1})} \\ &\cdot \left[1-(-1)^x\Big(\frac{1-2z_0-2w(1-z_1-z_0)}{1-2z_0-2w(z_1-z_0)}\Big)^{d_c-1}\right]. \end{aligned}$$

[3]For example, when $w = \frac{1}{2}$, $p_0^{(0)} = p_0^{(\ell)} = 8 \times 10^{-3}$, $p_1^{(0)} = p_1^{(\ell)} = 5 \times 10^{-3}$, $\alpha_0 = \beta_0 = 3 \times 10^{-3}$, $\alpha_1 = \beta_1 = 1 \times 10^{-3}$, $q_v = q_c = 10^{-3}$ and $d_v = 3$, $d_c = 6$ suffices for the approximation error of (8) to be within 8.3×10^{-5}.

For $w \in (0,1)$, and $(\frac{1}{2}-z_0)(\frac{1}{2}-z_1) > 0$, one can verify that $|\frac{1-2z_0-2w(1-z_1-z_0)}{1-2z_0-2w(z_1-z_0)}| < 1$, and $|1-2w| < 1$. Thus, when d_c is large enough, $\left(\frac{1-2z_0-2w(1-z_1-z_0)}{1-2z_0-2w(z_1-z_0)}\right)^{d_c-1} \approx 0$, $(1-2w)^{d_c-1} \approx 0$, and hence

$$\begin{aligned}\hat{\phi}_{Cx}(z_0,z_1) \approx& \frac{1}{2} - \frac{\big(1-2z_0-2w(z_1-z_0)\big)^{d_c-1}}{2} \\ =& \hat{\phi}_C((1-w)z_0+wz_1).\end{aligned}$$

Thus, (8) holds if at all iterations $\ell \geq 0$, $(\frac{1}{2}-p_0^{(\ell)})(\frac{1}{2}-p_1^{(\ell)}) > 0$. This is easily satisfied when all the error rate parameters specified in Section II are sufficiently small.

Now let $\bar{\alpha}_w = (1-w)\alpha_1 + w\alpha_0$, $\bar{\beta}_w = (1-w)\beta_1 + w\beta_0$, and $\bar{\gamma}_w = (1-w)\gamma_1 + w\gamma_0$. The following Theorem 2 gives the residual error rate of the Gallager B decoder when a codeword of relative weight w is transmitted.

Theorem 2. *When d_c is sufficiently large, if the error rates specified in Section II are sufficiently small, the residual error rate of the Gallager B decoder subject to both transient and permanent errors in message propagation is*

$$p_e \approx \begin{cases} \bar{\alpha}_w + q_v & d_v > 3, \\ \frac{\bar{\alpha}_w + q_v + 2\epsilon(\bar{\beta}_w + q_c)}{1-2(d_c-1)(\bar{\gamma}_w + (1-\gamma)\epsilon)} & d_v = 3. \end{cases}$$

Proof: The residual error rate is derived by finding the fixed point (p_0, p_1) of the recursive equation (8). We assume that condition (4) for the voting thresholds $b_x(z)$ to equal $\lceil \frac{d_v}{2} \rceil$ is satisfied for $z = p_x^{(\ell)}$ as the iteration number $\ell \to \infty$. We use linear approximations of functions $\hat{\phi}_C(z)$ and $\hat{\phi}_{Vx}(z)$. In addition, the cross-terms between the decoder circuit error rates q_v, q_c, α_0, α_1, β_0, β_1, γ_0, and γ_1 are dropped during the derivation. Details are omitted due to space constraints. ■

IV. Detecting Permanent Errors

In this section, we propose and evaluate a simple permanent error detection scheme that exploits the original decoder structure. As is observed from the expressions in Theorem 2, the transient errors at variable nodes and the permanent errors in (the memory cells storing) the variable-to-check messages are the two major components of the intrinsic residual error associated with our decoder. As proposed in [11], the effect of transient errors can be minimized by proper assignment of processors across different decoder components. Here, we further reduce the residual error rate through the detection of the permanent errors in variable-to-check messages. For convenience, in this section, the scope of the term "permanent error" is restricted to the permanent errors in variable-to-check messages.

We first classify the permanent errors into two categories, *perceptible* errors and *imperceptible* errors, depending on whether the stuck-at value is different from the associated variable node bit value or not. We note that the imperceptible errors cannot be identified because the stuck-at value coincides with the true value. However, since the positions of 0s and 1s may change as the transmitted codeword changes, an imperceptible error may become perceptible in subsequent transmissions (likewise perceptible error may become imperceptible; our detection scheme aims to identify perceptible errors before they become imperceptible).

We design a detection scheme that is performed in rounds. Each round starts with the transmission of a codeword, followed by a decoding phase, and finishes with a detection phase in which memory cells with perceptible permanent errors are detected and replaced with backup cells. The undetected perceptible errors and the imperceptible errors then constitute the *undiscovered errors* of the current round. They remain as the major source of residual errors of the next decoding phase, and are left to be detected and eliminated in the next detection phase.

A. Permanent Error Detection Scheme

Next, we describe the detection scheme performed in the detection phase of each round. Let $\{\Lambda_{v\to c}^{(\ell)}\}$ and $\{\Lambda_{c\to v}^{(\ell-1)}\}$, respectively, be the collection of variable-to-check messages and the collection of check-to-variable messages at the final iteration, say ℓ, of the decoding phase. We assume that ℓ is large enough so that the empirical residual decoding error is close enough to the residual error specified in Theorem 2.

To start the detection phase, each variable node v selects one of its d_v outgoing messages, referred to as θ_v, and sends θ_v to *all* of its neighboring check nodes. Each check node then XORs all the d_c messages coming from all of its d_c variable node neighbors. If, for a variable node v, none of its neighboring check nodes is satisfied, the location of the memory cell storing θ_v is recorded in the set Γ of error candidates. The procedure is repeated d_v times. At each time, each of the n variable nodes selects a new message among its d_v outgoing messages, and the set Γ is expanded accordingly. Next, to suppress the effects of transient errors, we take $\{\Lambda_{c\to v}^{(\ell-1)}\}$ as the input, and repeat the computations at the variable nodes. Each newly computed v-to-c message overwrites $\Lambda_{v\to c}^{(\ell)}$, and reads as $\tilde{\Lambda}_{v\to c}^{(\ell)}$ when accessed from the memory cell. Note that $\{\tilde{\Lambda}_{v\to c}^{(\ell)}\}$ and $\{\Lambda_{v\to c}^{(\ell)}\}$ are the same except for the messages that experience only transient errors. We then repeat the same procedure previously performed on $\{\Lambda_{v\to c}^{(\ell)}\}$: based on $\tilde{\Lambda}_{v\to c}^{(\ell)}$ we identify variable nodes with all checks unsatisfied, and derive the set $\tilde{\Gamma}$ from $\{\tilde{\Lambda}_{v\to c}^{(\ell)}\}$. Finally, the memory cells recorded in $\Gamma \cap \tilde{\Gamma}$ are labeled as having permanent errors, and are replaced. By taking the intersection of the two candidate error sets, the risk of mislabeling a transient error as a permanent error is significantly reduced.

B. Performance Analysis

It remains to evaluate the permanent error detection capability of the proposed scheme. We measure the scheme performance by the undiscovered error rate $P_{\mathrm{ud},k}$ at the end of round k. Recall that the undiscovered errors are composed of the imperceptible errors and the undetected perceptible errors.

Prior to the first round, we have $P_{\mathrm{ud},0} = \alpha = \alpha_0 + \alpha_1$. Let ρ_k be the perceptible permanent error rate at round k. At round $k = 1$, we have $\rho_1 = \bar{\alpha}_{w_1}$, where $\bar{\alpha}_w = (1-w)\alpha_1 + w\alpha_0$ as defined in Section III, and w_1 is the relative weight of the first transmitted codeword. Note that a perceptible permanent error will not be detected if at least one neighboring check for the variable node of interest is satisfied. Thus, define

$$P_M(u) \quad = 1 - \Big[1 - (q_c + (1-2q_c)\hat{\phi}_C(u+q_v))\Big]^{d_v}$$

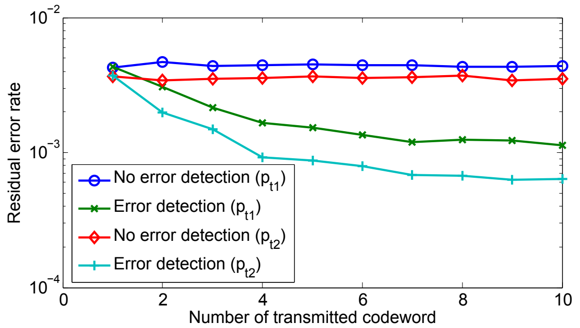

Fig. 1. Performance comparison, permanent error rate 5×10^{-3}, transient error rate $p_{t1} = 10^{-3}$, $p_{t2} = 5 \times 10^{-4}$.

as a function of $u \in [0, 1]$ (the function $\hat{\phi}_C(z)$ is defined in (2)). Given that the perceptible permanent error rate is ρ_k, $P_M(\rho_k)$ is an approximation of the probability that a perceptible permanent error is not detected. The approximation error is small when q_v and ρ_k are sufficiently small. Since $\hat{\phi}_C(\rho_k + q_v) \approx 1$, we have $P_M(\rho_k) \approx 0$ when q_c is sufficiently small. Then, we are able to find by induction (details omitted due to space constraints) that

$$\begin{aligned} P_{\text{ud},k} &= P_{\text{ud},k,0} + P_{\text{ud},k,1} \qquad (10) \\ &= \alpha_0 \prod_{i=1}^{k} \left[w_i P_M(\rho_i) + (1 - w_i) \right] \\ &+ \alpha_1 \prod_{i=1}^{k} \left[(1 - w_i) P_M(\rho_i) + w_i \right], \end{aligned}$$

where $\rho_i = w_i P_{\text{ud},i,0} + (1 - w_i) P_{\text{ud},i,1}$, and w_i is the relative weight of the ith transmitted codeword. Note that $P_{\text{ud},k,0}$ is in fact the product of the undiscovered stuck-at-0 permanent error rates in all k rounds, and $P_{\text{ud},k,1}$ is the product of the undiscovered stuck-at-1 permanent error rates in all k rounds.

It can be shown (similar to the derivation of $P_M(u)$) that the probability of mislabeling functional memory cells as being defective is roughly q_v^2, which is negligible for small q_v.

C. Simulation results

We ran a (3,6)-regular LDPC code with codelength 2640 from [15] on the noisy LDPC decoders with $\alpha_1 = \beta_1 = \gamma_1 = 3 \times 10^{-3}$ and $\alpha_0 = \beta_0 = \gamma_0 = 2 \times 10^{-3}$. The overall permanent error rate was 5×10^{-3}. We considered the following values for the transient error rates: $q_v = q_c = p_{t1} = 10^{-3}$ and $q_v = q_c = p_{t2} = 5 \times 10^{-4}$. The channel error was $\epsilon = 5 \times 10^{-3}$. We compared the noisy decoder incorporating the proposed detection scheme with the nominal noisy decoder (without detection). The simulation results shown in Fig. 1 reveal the improvement in the performance of the noisy decoder when we deploy the proposed detection scheme. As more codewords are transmitted, more permanent errors are detected and the performance of the noisy decoder continually improves. In our example, after 7 codeword are transmitted, the residual error rate drops close to the transient error rate, whereas the nominal noisy decoder residual error rate is still the sum of the transient error rate and the perceptible permanent error rate.

V. Conclusion and Future Work

In this paper, we studied a noisy Gallager B decoder implemented on hardware built out of processors with transient errors and memory units with permanent errors. We derived both the exact and approximate density evolution expressions for the decoder output bit error rate as a function of channel noise and transient and permanent decoder errors. Additionally, we proposed a scheme for detecting permanent errors and we illustrated the method with an accompanying example. We also provide additional simulation results and examples in [16]. As a measure of the undesirable cost incurred from replacing memory cells, the probability of mislabeling functional memory cells as being defective is analyzed in the extended version of this paper [16]. Now that we have characterized the effectiveness of the proposed permanent-error detection scheme, we will further be able to evaluate the cost-effectiveness of this and other on-circuit error mitigation mechanisms (such as the resource assignment scheme in [11]), provided that proper cost models for employing these mechanisms can be established. Future work may involve channel code optimization techniques that take into account both transmission as well as (possibly asymmetric) processing errors, and possible applications to other inference algorithms implemented on noisy circuits.

Acknowledgment

This work was supported by NSF Grant CCF-1029030.

References

[1] J. von Neumann, "Probabilistic logic and the synthesis of reliable organisms from unreliable components," *Automata Studies*, pp. 43–98, 1956.
[2] J.-Y. Jou and J. Abraham, "Fault-tolerant matrix arithmetic and signal processing on highly concurrent computing structures," *Proc. of the IEEE*, vol. 74, no. 5, pp. 732 – 741, May 1986.
[3] C. Hadjicostis and G. Verghese, "Coding approaches to fault tolerance in linear dynamic systems," *IEEE Trans. Inform. Theory*, vol. 51, no. 1, pp. 210 – 228, Jan. 2005.
[4] M. G. Taylor, "Reliable information storage in memories designed from unreliable components," *Bell Syst. Tech. J.*, vol. 47, no. 10, pp. 2299–2337, Dec. 1968.
[5] C. Heegard, "Partitioned linear block codes for computer memory with 'stuck-at' defects," *IEEE Trans. Inform. Theory*, vol. 29, no. 6, pp. 831 – 842, Nov. 1983.
[6] L. Lastras-Montaño, A. Jagmohan, and M. M. Franceschini, "Algorithms for memories with stuck cells," in *IEEE ISIT*, Jun. 2010.
[7] L. R. Varshney, "Performance of LDPC codes under faulty iterative decoding," *IEEE Trans. Inform. Theory*, vol. 57, no. 7, pp. 4427–4444, Jul. 2011.
[8] F. Leduc-Primeau and W. J. Gross, "Faulty gallager-b decoding with optimal message repetition," in *Allerton Conference*, Oct. 2012.
[9] A. Tarighati, H. Farhadi, and F. Lahouti, "Performance analysis of noisy message-passing decoding of low-density parity-check codes," in *IEEE ISTC*, Sep. 2010.
[10] S. M. S. Tabatabaei, H. Cho, and L. Dolecek, "Gallager B decoder on noisy hardware," *IEEE Trans. Commun.*, 2013, accepted.
[11] S. M. S. Tabatabaei, C.-H. Huang, and L. Dolecek, "Optimal design of a Gallager B noisy decoder for irregular LDPC codes," *IEEE Commun. Lett.*, vol. 16, no. 12, pp. 2052 –2055, Dec. 2012.
[12] R. G. Gallager, "Low-Density Parity-Check Codes," *Cambridge, MA: MIT Press*, 1963.
[13] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.
[14] C.-C. Wang, S. Kulkarni, and H. Poor, "Density evolution for asymmetric memoryless channels," *IEEE Trans. Inform. Theory*, vol. 51, no. 12, pp. 4216 –4236, Dec. 2005.
[15] D. MacKay, "Encyclopedia of sparse graph codes," *Available online: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html*.
[16] C.-H. Huang, Y. Li, and L. Dolecek, "Gallager B LDPC decoder with transient and permanent errors," *IEEE Trans. Commun.*, 2013, submitted.