Min-Sum-based decoders running on noisy hardware

Christiane Kameni Ngassa*†, Valentin Savin*, David Declercq†
*CEA-LETI, MINATEC Campus, 17 rue des Martyrs, 38054 Grenoble, France
{christiane.kameningassa, valentin.savin}@cea.fr
†ETIS ENSEA/UCP/CNRS UMR 8051, 95014 Cergy-Pontoise Cedex, France
declercq@ensea.fr

***Abstract*—This paper deals with Low-Density Parity-Check decoders running on noisy hardware. This represents an unconventional paradigm in communication theory, since it is traditionally assumed that the error correction decoder operates on error-free devices and the randomness (in the form of noise and/or errors) exists only in the transmission channel. However, with the advent of nanoelectronics, it starts to be widely accepted that the future generations of circuits and systems will need to reliability compute and solve statistical inferences, by making use of unreliable "noisy" components. It is then critical to properly evaluate the robustness of the existing decoders in the presence of an additional source of noise at the circuit level. To this end, we first introduce a new error model approach and carry out the "noisy" density evolution analysis of the fixed-point Min-Sum decoding. Then, for different parameters of the noisy components of the decoder, we determine the range of the signal-to-noise ratio values for which the decoder is able to achieve a target bit error rate performance. Finally, we evaluate the finite-length performance of the Min-Sum and two other Min-Sum-based decoders running on noisy hardware.**

I. Introduction

In traditional models of communication systems with error correction coding, it is assumed that the operations of an error correction encoder and decoder are deterministic and that the randomness exists only in the transmission channel. However, with the advent of nanoelectronics, the reliability of the forthcoming circuits and computation devices is becoming questionable. Indeed, due to huge increases in density integration, lower supply voltages, and variations in the technological process, MOS and emerging nanoelectronic devices will be inherently unreliable. Besides, a significant challenge to current CMOS design is to lower the energy consumption by several factors of magnitude, with the obvious goal of energy preservation. Diminishing the energy consumption can be addressed by *aggressive supply voltage scaling*, with the drawback that bringing the signal level closer to the noise level reduces noise immunity and leads to unreliable computing. It is then becoming crucial to design and analyze error correcting decoders able to provide reliable error correction even if they are made of unreliable components.

Except the pioneered works by Taylor and Kuznetsov on reliable memories [1]–[3], later generalized in [4], [5] to the case of hard-decision decoders, this new paradigm of noisy decoders has merely not been addressed until recently in the coding literature. However, over the last years, the study of error correcting decoders, especially Low-Density Parity-Check (LDPC) decoders, running on noisy hardware attracted more and more interest in the coding community. In [6] and [7] hardware redundancy is used to develop fault-compensation techniques, able to protect the decoder against the errors induced by the noisy components of the circuit. In [8], a class of modified Turbo and LDPC decoders has been proposed, able to deal with the noise induced by the failures of a low-power buffering memory that stores the input soft bits of the decoder. Very recently, the characterization of the effect of noisy logic components (faulty gates) on standard iterative LDPC decoders has been proposed. In [9], the concentration and convergence properties were proved for the asymptotic performance of noisy message-passing decoders, and density evolution equations were derived for the noisy Gallager-A and Belief-Propagation decoders. In [10]–[12], the authors investigated the asymptotic behavior the noisy Gallager-B decoder defined over binary and non-binary alphabets. However, all these papers deal with very simple error models, which emulate the noisy implementation of the decoder, by passing each of the exchanged messages through a noisy channel.

In this paper we focus on the Min-Sum and Min-Sum-based decoders, which are widely implemented in real communication systems. In order to emulate the noisy implementation of the decoder, probabilistic error models are proposed for its arithmetic components (adders and comparators). The proposed probabilistic components are used to build the noisy fixed-point decoders. We further analyze the asymptotic performance of the noisy Min-Sum decoder, and provide useful regions and target-BER-thresholds [9] for a wide range of parameters of the proposed error models. Finally, we investigate the finite length performance of the noisy Min-Sum, Offset-Min-Sum, and Self-Corrected Min-Sum decoders.

II. LDPC Codes and Min-Sum Algorithm

A. LDPC Codes

LDPC codes [13] are linear block codes defined by sparse parity-check matrices. They can be advantageously represented by bipartite (Tanner) graphs [14] and decoded by message-passing iterative algorithms. The Belief-Propagation (BP) decoding – also referred to as Sum-Product (SP) – is known to be optimal on cycle-free graphs, but can also be successfully applied to decode linear codes defined by graphs with cycles, which is actually the case of all practical codes. However, in practical applications the BP algorithm is disadvantaged by

its computational complexity and the fact that it requires the perfect knowledge of the channel parameter (*e.g.* SNR), which may be imprecisely estimated in practical situations.

The Min-Sum (MS) algorithm is aimed at reducing the computational complexity of the BP, by using max-log approximations of the parity-check to coded-bit messages. The only computations required by the MS decoding are additions and comparisons, which solves the computational complexity and numerical instability problems. For most of the usual channel models, the performance of the MS decoding is also known to be independent of the knowledge of the channel parameter.

B. Notations

We consider an LDPC code defined by a bipartite (Tanner) graph $\mathcal{H}$, with N variable-nodes and M check-nodes [14]. Variable-nodes and check-node are denoted, respectively, by $n \in \{1, 2, ..., N\}$ and $m \in \{1, 2, ..., M\}$. $\mathcal{H}(n)$ and $\mathcal{H}(m)$ denote the set of neighbor nodes of the variable-node n and of the check-node m, respectively.

We further consider a codeword $(x_1, \ldots, x_N)$ that is sent over a memoryless noisy channel, and denote by $(y_1, \ldots, y_N)$ the received word. The following notation will be used throughout the paper, with respect to message passing decoders:

- γ_n is the log-likelihood ratio (LLR) value of x_n according to the received y_n value; it is also referred to as the *a priori information* of the decoder concerning the variable-node n;
- $\tilde{\gamma}_n$ is the *a posteriori information* (LLR value) of the decoder concerning the variable-node n;
- $\alpha_{m,n}$ is the variable-to-check message sent from variable-node n to check-node m;
- $\beta_{m,n}$ is the check-to-variable message sent from check-node m to variable-node n.

C. Min-Sum Decoding

The MS decoding works as follows. First variable to-check-messages are initialized according to the corresponding a priori LLR values. Then each decoding iteration consists of three steps, namely the check-node (CN) processing step, the a posteriori (AP) information update, and the variable-node (VN) processing step.

Initialization

- $\gamma_n = \log\left(\dfrac{\Pr(x_n = 0|y_n)}{\Pr(x_n = 1|y_n)}\right)$, $\forall\ n \in \{1, \ldots, N\}$;
- $\alpha_{m,n} = \gamma_n$, $\forall\ n \in \{1, \ldots, N\}$ and $m \in \mathcal{H}(n)$;

Iterations

- CN-processing: $\forall\ m \in \{1, \ldots, M\}$ and $n \in \mathcal{H}(m)$

$$\beta_{m,n} = \Big(\prod_{n' \in H(m)\setminus n} \operatorname{sgn}(\alpha_{m,n'})\Big) \min_{n' \in H(m)\setminus n} (|\alpha_{m,n'}|)$$

- AP-update: for $\forall\ n \in \{1, \ldots, N\}$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

- VN-processing: for $\forall\ n \in \{1, \ldots, N\}$ and $m \in \mathcal{H}(n)$

$$\alpha_{m,n} = \tilde{\gamma}_n - \beta_{m,n}$$

III. Noisy Min-Sum Decoder

In order to emulate the noisy implementation of the Min-Sum decoders, we propose realistic error models for adders and the comparators, which are the only two arithmetic components of the decoder. We shall not consider here the case of errors that may occur due to the temporary storage of the exchanged messages in possibly noisy memories (in order not to depend on a specific decoder architecture). However, we note that the effect of noisy memories on the exchanged messages can be integrated into the probabilistic models of the arithmetic components, in the sens that adding noise in memories would modify the parameters of the probabilistic arithmetic components.

As our goal is to investigate a Min-sum decoder implemented on noisy hardware, we have to consider a fixed-point (quantized) Min-sum decoder. The number of quantization bits used for the a priori / a posteri information and exchanged messages will also determine the number of bits of its fixed-point arithmetic components.

A. Fixed-Point Min-Sum decoder

We consider a fixed-point Min-Sum decoder, in which the a priori information (γ_n) and the exchanged messages ($\alpha_{m,n}$ and $\beta_{m,n}$) are quantized on q bits. The a posteriori information $\tilde{\gamma}_n$ is quantized on $\tilde{q}$ bits with $\tilde{q} > q$ (usually $\tilde{q} = q + 1$, or $\tilde{q} = q + 2$). We further denote:

- $\mathcal{M} = \{-Q, \ldots, -1, 0, +1, \ldots, +Q\}$, where $Q = 2^{q-1} - 1$, the alphabet of the a priori information and of the exchanged messages;
- $\widetilde{\mathcal{M}} = \{-\widetilde{Q}, \ldots, -1, 0, +1, \ldots, +\widetilde{Q}\}$, where $Q = 2^{\tilde{q}-1} - 1$, the alphabet of the a posteriori information;
- $\mathbf{q} : \mathbb{R} \to \mathcal{M}$, a fixed q-bit quantization map
- $\mathbf{s}_{\mathcal{M}} : \mathbb{Z} \to \mathcal{M}$, the q-bit saturation map defined by:

$$\mathbf{s}_{\mathcal{M}}(z) = \begin{cases} -Q, & \text{if } z < -Q \\ z, & \text{if } z \in \mathcal{M} \\ +Q, & \text{if } z > +Q \end{cases}$$

- $\mathbf{s}_{\widetilde{\mathcal{M}}} : \mathbb{Z} \to \widetilde{\mathcal{M}}$, the $\tilde{q}$-bit saturation map defined in a similar manner as the previous one

The quantization map $\mathbf{q}$ determines the q-bit quantization of the a priori LLR values, while saturation maps $\mathbf{s}_{\mathcal{M}}$ and $\mathbf{s}_{\widetilde{\mathcal{M}}}$ define the fixed-point saturation of the exchanged messages and a posteriori LLR values.

B. Model for the Noisy Adder

Adders are used in the decoder to compute the a posteriori information $\tilde{\gamma}_n$ (quantized on $\tilde{q}$ bits) and messages $\alpha_{m,n}$ (quantized on q bits). Given that $\tilde{q} > q$, we only consider $\tilde{q}$-bits adders (which also corresponds to practical implementations, since the value of $\alpha_{m,n}$ is derived from that of $\tilde{\gamma}_n$).

The noisy (probabilistic) adder is defined by the following parameters:

- p_a is the probability that the adder's output is in error;
- q_e is the number of bits of the adder output, starting from the least significant bit (LSB), that can be affected by errors. Hence, $q_e \leq \tilde{q}$, and it is referred as the *depth* of the probabilistic model.

Table I
EXAMPLE OF AN ERROR INJECTION IN THE OUTPUT OF THE NOISY ADDER

	integer	2's complement bit representation				
exact output	-11	1	0	1	0	1
error pattern	6		0	1	1	0
erroneous output	-13	1	0	0	1	1
bit position		$\tilde{q}=5$	$q_e=4$	3	2	1

The probabilistic model is further specified as follows. Let d be an integer, referred to as error-pattern, for which the positions of 1's (within its binary representation) indicate the locations of the erroneous bits in the adder's output. The output of the adder is error-free if and only if $d=0$. The error-pattern d belongs to an alphabet $\mathcal{D}$, which is defined as follows.

- When $q_e < \tilde{q}$ the error-pattern d is an unsigned integer represented on q_e-bits, hence $\mathcal{D}=\{0,1,\ldots,Q_e\}$, with $Q_e = 2^{q_e}-1$.
- When $q_e = \tilde{q}$ the sign of the output may also be in error, hence d is a signed integer represented on $q_e=\tilde{q}$ bits and $\mathcal{D}=\{-\widetilde{Q},\ldots,-1,0,1,\ldots,\widetilde{Q}\}$

The output of the noisy adder is obtained by performing a bit-wise xor operation between the output of the noiseless adder and d. Furthermore, we consider that all non-zero error patterns have equal probability. Since $p_a = \sum_{d\neq 0}\Pr(d)$, it follows that for any $d\neq 0$, $\Pr(d) = \frac{p_a}{|\mathcal{D}|-1}$, while $p(0)=1-p_a$.

Finally, the probabilistic adder is defined by:

$$\mathbf{a}_{\text{pr}}(x,y) = \mathbf{s}_{\widetilde{\mathcal{M}}}(x+y)\wedge d,\ \forall (x,y)\in\widetilde{\mathcal{M}}^2,$$

where d is drawn randomly from $\mathcal{D}$ according to the above probabilities, and $\wedge$ symbol denotes the bitwise xor operation.

Table I gives an example of an erroneous adder output, for $\tilde{q}=5$ and $q_e=4$. In this case, since $q_e<\tilde{q}$, the sign bit of the output cannot be affected by errors. However, it worth noting that in case of an "addition chain", as for instance $(x+y)+z$, the sign of the noiseless fixed-point output, given by $\mathbf{s}_{\widetilde{\mathcal{M}}}(\mathbf{s}_{\widetilde{\mathcal{M}}}(x+y)+z)$, may be different from the sign of the noisy output, given by $\mathbf{a}_{\text{pr}}(\mathbf{a}_{\text{pr}}(x,y),z)$.

Finally, we note that the depth parameter q_e is used to investigate the decoder behavior when errors may occur on increasing significantly bits. It can be used as a *guideline* for the hardware architecture of adders made from noisy components (logic gates). In order to ensure a target performance of the decoder, *adders should be specifically designed, such that to be compliant with the maximum admissible q_e value* (e.g. by using classical fault-tolerant solutions for the MSBs, as modular redundancy).

C. Model for Noisy Comparator

The probabilistic error model of the noisy comparator is simpler, and it is specified only by a single error probability parameter, denoted by p_c. We consider q-bit comparators that are used at the CN-processing step, and for any $x,y\in\mathcal{M}$ the probabilistic **minimum** of x and y, denoted by $\mathbf{m}_{\text{pr}}(x,y)$ is defined by:

$$\mathbf{m}_{\text{pr}}(x,y) = \begin{cases} \min(x,y), & \text{with probability } 1-p_c \\ \max(x,y), & \text{with probability } p_c \end{cases}$$

D. Quantized Noisy Min-Sum Decoding

Using the previous notation, the noisy fixed-point Min-Sum decoder can be described as follows (probabilistic components appear in red):

Initialization

- $\gamma_n = \mathbf{q}\left[\ln\left(\frac{\Pr(x_n=0\mid y_n)}{\Pr(x_n=1\mid y_n)}\right)\right],\ \forall\ n\in\{1,\ldots,N\}$;
- $\alpha^{(0)}_{m,n}=\gamma_n,\ \forall\ n\in\{1,\ldots,N\}$ and $m\in\mathcal{H}(n)$;

Iterations: for $\ell\geq 1$

- CN-processing: $\forall\ m\in\{1,\ldots,M\}$ and $n\in\mathcal{H}(m)$

$$\beta^{(\ell)}_{m,n} = \prod_{i=1,\ldots,d_c-1}\text{sgn}\left(\alpha^{(\ell-1)}_{m,n_i}\right)\cdot \mathbf{m}_{\text{pr}}\left(\mathbf{m}_{\text{pr}}\left(|\alpha^{(\ell-1)}_{m,n_1}|,|\alpha^{(\ell-1)}_{m,n_2}|\right)\cdots,|\alpha^{(\ell-1)}_{m,n_{d_c-1}}|\right)$$

- AP-update: for $\forall\ n\in\{1,\ldots,N\}$

$$\tilde{\gamma}^{(\ell)}_n = \mathbf{a}_{\text{pr}}\left(\mathbf{a}_{\text{pr}}\left(\gamma_n,\beta^{(\ell)}_{m_1,n}\right)\cdots,\beta^{(\ell)}_{m_{d_v},n}\right)$$

- VN-processing: for $\forall\ n\in\{1,\ldots,N\}$ and $m\in\mathcal{H}(n)$

$$\alpha^{(\ell)}_{m,n} = \mathbf{a}_{\text{pr}}\left(\tilde{\gamma}^{(\ell)}_n,-\beta^{(\ell)}_{m,n}\right)$$

We note that the CN processing step may induce errors on the absolute value of check-to-variable messages, but the sign computation is performed in a reliable (deterministic) way. The reason is that errors occurring the sign of exchanged messages drastically degrade the decoder performance. Since the circuitry to compute the sign of check-to-variable messages is very simple, we can reasonably assume that the sign is reliably computed, for instance by using modular redundancy, or multi-voltage design techniques that increase the supply voltage of the sign circuit.

IV. DENSITY EVOLUTION EQUATIONS FOR NOISY MIN-SUM DECODING

In this section we derive density evolution equations for the noisy fixed-point MS decoding for a regular (d_v,d_c) LDPC code. The study can be easily generalized to irregular LDPC codes, simply by averaging according to the *edge perspective* degree distribution polynomials.

The objective of the density evolution technique is to recursively compute the probability mass functions of exchanged messages, through the iterative decoding process. This is done under the independence assumption of exchanged messages, holding in the asymptotic limit of the code length, in which case the decoding performance converges to the cycle-free case [9]. Due to the symmetry of the decoder, the analysis can be further simplified by assuming that the all-zero codeword is transmitted through the channel. We note that our analysis applies to any memoryless symmetric channel. The following notation will be used throughout this section.

- $C(z)=\Pr(\gamma=z),\forall z\in\mathcal{M}$
- $\widetilde{C}^{(\ell)}(\tilde{z})=\Pr(\tilde{\gamma}^{(\ell)}=\tilde{z}),\forall\tilde{z}\in\widetilde{\mathcal{M}}$
- $A^{(\ell)}(z)=\Pr\left(\alpha^{(\ell)}=z\right),\forall z\in\mathcal{M}$
- $B^{(\ell)}(z)=\Pr\left(\beta^{(\ell)}=z\right),\forall z\in\mathcal{M}$

The decoder's error probability at iteration ℓ, is defined by:

$$P_e^{(\ell)} = \sum_{\tilde{z}=-\widetilde{Q}}^{0} \widetilde{C}^{(\ell)}(\tilde{z})$$

In order to be able to recursively compute $A^{(\ell)}(z), B^{(\ell)}(z)$, and $\widetilde{C}^{(\ell)}$, for $\ell > 0$, the VN-processing step must be further modified as follows:

- VN-processing: for $\forall\ n \in \{1, \ldots, N\}$ and $m \in \mathcal{H}(n)$

$$\alpha_{m,n}^{(\ell)} = \mathbf{a}_{\text{pr}}\left(\mathbf{a}_{\text{pr}}\left(\gamma_n + \beta_{m_1,n}^{(\ell)}\right) \cdots + \beta_{m_{d_v-1},n}^{(\ell)}\right)$$
$$\alpha_{m,n}^{(\ell)} = \mathbf{s}_{\mathcal{M}}\left(\alpha_{m,n}^{(\ell)}\right)$$

This modification is needed in order to avoid computing $\alpha_{m,n}^{(\ell)} = \mathbf{a}_{\text{pr}}\left(\tilde{\gamma}_n^{(\ell)}, -\beta_{m,n}^{(\ell)}\right)$, since it involves two correlated variables, namely $\tilde{\gamma}_n^{(\ell)}$ and $\beta_{m,n}^{(\ell)}$. For floating-point noiseless decoders, the two ways of computing the variable-to-check messages are completely equivalent. However, this equivalence does not hold any more for noisy decoders (it does not actually hold even for noiseless, but fixed-point, decoders!).

A. Expression of $B^{(\ell)}$ as a function of $A^{(\ell-1)}$

In the sequel, we make the convention that $\Pr(\text{sgn}(0) = 1) = \Pr(\text{sgn}(0) = -1) = 1/2$. The followng notation will be used:

- $A_{[x,y]} = \sum_{z=x}^{y} A(z)$, for $x \leq y \in \mathcal{M}$
- $A_{[0^+,y]} = \frac{1}{2}A(0) + \sum_{z=1}^{y} A(z)$, for $y \in \mathcal{M},\ y > 0$
- $A_{[x,0^-]} = \frac{1}{2}A(0) + \sum_{z=x}^{-1} A(z)$, for $x \in \mathcal{M},\ x < 0$

For the sake of simplicity, we drop the iteration index, thus $B := B^{(\ell)}$ and $A := A^{(\ell-1)}$. We proceed by recursion on $i = 2, \ldots, d_c - 1$, where d_c denotes the check-node degree. Let $\beta_1 := \alpha_1$, and for $i = 2, \ldots, d_c - 1$ define:

$$\beta_i = \text{sgn}(\beta_{i-1})\text{sgn}(\alpha_i)\mathbf{m}_{\text{pr}}(|\beta_{i-1}|, |\alpha_i|)$$

Let B_{i-1} and B_i be the probability mass functions of β_{i-1} and β_i, respectively (hence, $B_1 = A$). We have:

For $z = 0$,
$$B_i(0) = \Pr(\beta_i = 0) = A(0)B_{i-1}(0) + [B_{i-1}(0)(1 - A(0)) + A(0)(1 - B_{i-1}(0))]\,(1 - p_c)$$

For $z > 0$,
$$F_i(z) \stackrel{\text{def}}{=} \Pr(\beta_i \geq z) = \left[B_{i-1}{}_{[0^+,z-1]}A_{[z,Q]} + A_{[0^+,z-1]}B_{i-1}{}_{[z,Q]}\right] p_c + \left[B_{i-1}{}_{[1-z,0^-]}A_{[-Q,-z]} + A_{[1-z,0^-]}B_{i-1}{}_{[-Q,-z]}\right] p_c + B_{i-1}{}_{[z,Q]}A_{[z,Q]} + B_{i-1}{}_{[-Q,-z]}A_{[-Q,-z]}$$
$$B_i(z) = \Pr(\beta_i = z) = F_i(z) - F_i(z+1)$$

For $z < 0$,
$$G_i(z) \stackrel{\text{def}}{=} \Pr(\beta_i \leq z) = \left[B_{i-1}{}_{[0^+,-z-1]}A_{[-Q,z]} + A_{[0^+,-z-1]}B_{i-1}{}_{[-Q,z]}\right] p_c + \left[B_{i-1}{}_{[-z,Q]}A_{[z+1,0^-]} + A_{[-z,Q]}B_{i-1}{}_{[z+1,0^-]}\right] p_c + B_{i-1}{}_{[-z,Q]}A_{[-Q,z]} + A_{[-z,Q]}B_{i-1}{}_{[-Q,z]}$$
$$B_i(z) = \Pr(\beta_i = z) = G_i(z) - G_i(z-1)$$

Finally, we have that $B = B_{d_c-1}$.

B. Expression of $A^{(\ell)}$ as a function of $B^{(\ell)}$ and C

We also derive in same time the expression of $\widetilde{C}^{(\ell)}$ as a function of $B^{(\ell)}$ and C. We drop the iteration index for simplicity (so $A := A^{(\ell)}$, $B := B^{(\ell)}$, and $\widetilde{C} = \widetilde{C}^{(\ell)}$) and we proceed by recursion on $i = 0, 1, \ldots, d_v$, where d_v denotes the variable-node degree. We denote by $\wedge$ the bitwise xor operator, and further define:

- $\Omega_0 = \gamma \in \mathcal{M} \subseteq \widetilde{\mathcal{M}}$, $C_0(\tilde{z}) = \Pr(\Omega_0 = \tilde{z})$
- for $i = 1, \ldots, d_v - 1$, $w \in \mathbb{Z}$, $\tilde{w} \in \widetilde{\mathcal{M}}$ and $\tilde{z} \in \widetilde{\mathcal{M}}$

$$\begin{cases} \omega_i = \Omega_{i-1} + \beta_{m_i,n} \in \mathbb{Z}, & c_i(w) = \Pr(\omega_i = w) \\ \tilde{\omega}_i = \mathbf{s}_{\widetilde{\mathcal{M}}}(\omega_i) \in \widetilde{\mathcal{M}}, & \tilde{c}_i(\tilde{w}) = \Pr(\tilde{\omega}_i = \tilde{w}) \\ \Omega_i = \tilde{\omega}_i \wedge d \in \widetilde{\mathcal{M}}, & \widetilde{C}_i(\tilde{z}) = \Pr(\Omega_i = \tilde{z}) \end{cases}$$

It follows that:

- $\widetilde{C}_0(\tilde{z}) = \begin{cases} C(\tilde{z}), & \text{if } \tilde{z} \in \mathcal{M} \\ 0, & \text{if } \tilde{z} \in \widetilde{\mathcal{M}} \setminus \mathcal{M} \end{cases}$
- for $i = 1, \ldots, d_v$,

$$\begin{cases} c_i(w) = \sum_u \widetilde{C}_{i-1}(u)B(w-u) \\ \tilde{c}_i = \mathbf{s}_{\widetilde{\mathcal{M}}}(c_i) \\ \widetilde{C}_i(\tilde{z}) = \sum_d \Pr(d)\tilde{c}_i(\tilde{z} \wedge d) \end{cases}$$

- $A = \mathbf{s}_{\mathcal{M}}\left(\widetilde{C}_{d_v-1}\right)$
- $\widetilde{C} = \widetilde{C}_{d_v}$

In the above equations, applying the saturation operator ($\mathbf{s}_{\widetilde{\mathcal{M}}}$ or $\mathbf{s}_{\mathcal{M}}$) on a probability mass function means that all the non-zero probabilities corresponding to values outside the alphabet ($\widetilde{\mathcal{M}}$ or $\mathcal{M}$) must be accumulated to the probability of the corresponding boundary value of the alphabet.

The decoder's error probability at the current (ℓ) iteration is given by $P_e = \sum_{\tilde{z}=-\widetilde{Q}}^{0} \widetilde{C}(\tilde{z})$. Finally, we note that the density evolution equations for the fixed-point noiseless Min-Sum decoder can be obtained by setting $p_a = 0$ and $p_c = 0$.

V. Numerical Results

A. Decoding thresholds for the noisy Min-Sum Decoder

Density evolution equations for noisy fixed-point MS decoder were run on MATLAB for a regular $(3, 6)$ LDPC code over the Binary-Input AWGN channel, with exchanged messages and a priori information quantized on $q = 4$ bits, and the a posteriori information quantized on $\tilde{q} = 5$ bits. Since the input error probability can actually be increased when the decoder is run on noisy hardware, the first step is to evaluate the channel and hardware parameters yielding a final probability of error (after decoding) less than the channel error probability. Following [9], decoder is said to be *useful* if the $\lim_{\ell\to\infty} P_e^{(\ell)}$ exists and:

$$\lim_{\ell\to\infty} P_e^{(\ell)} \leq p_0$$

where $P_e^{(\ell)}$ is the error probability at iteration ℓ, and p_0 is the error probability of a hard decision on the received sequence. The ensemble of the parameters that satisfy this condition constitutes the *useful region* of the decoder.

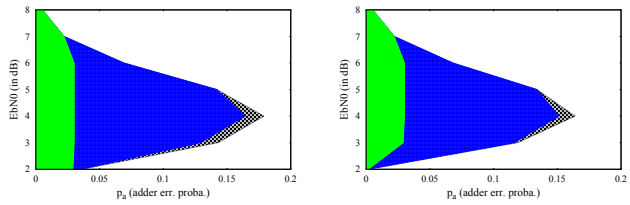
For noiseless-decoders traditionally considered in classical coding theory, the decoding threshold is defined as the maximum channel noise, such that the error probability converges to zero as the number of decoding iterations goes to infinity. However, for noisy decoders this error probability does not converge to zero, and an alternative definition of the decoding threshold has been introduced in [9]. Accordingly, for a target bit-error rate η, the η-threshold is defined by:

$$\sigma^*(\eta) = \sup\{\sigma : \lim_{\ell \to \infty} P_e^{(\ell)} < \eta\}$$

where σ is the standard deviation of the Gaussian noise.

Fig. 1 plots the useful regions for $p_c = 0.001$ and $p_c = 0.01$, and for depth $q_e = 4$ and $q_e = 5$. The blue and green regions are parts of the useful region for which BER $\leq 10^{-2}$ and BER $\leq 10^{-3}$, respectively. For more visibility, threshold values are displayed as E_b/N_0 values in decibels. The values considered for p_a and p_c are rather high with respect to the rate at which soft errors occur in current technologies. However, our goal is to investigate the possible behavior of the decoder for advanced technologies, or in case that *aggressive* voltage scaling is used as a technique to lower the energy consumption of the circuit (Section I).

From Fig. 1, one can see that the noise from the comparator has only a limited impact on the MS decoder performance, as the useful regions are almost the same when the error probability p_c is multiplied by 10. For $q_e = 4$, the maximum value of p_a that allows useful decoding is $p_a = 0.179$ for $p_c = 0.001$, and $p_a = 0.164$ for $p_c = 0.01$. For $q_e = 5$, the maximum p_a is 0.009 for both $p_c = 0.001$ and $p_c = 0.01$. Note that the useful region decreases severely when the depth increases from $q_e = 4$ to $q_e = 5$. This confirms the vulnerability of the MS decoder to the noise on the MSB bit of the adder. Fig. 1 indicates that in order to achieve a BER $\leq 10^{-3}$ we should have $p_a \leq 0.03$ for $q_e = 4$, and $p_a \leq 0.0012$ for $q_e = 5$. Theses results are consistent with the finite length simulation results that will be discussed in the next section.



(a) Depth $q_e = 4$, $p_c = 0.001$ (b) Depth $q_e = 4$, $p_c = 0.01$

(c) Depth $q_e = 5$, $p_c = 0.001$ (d) Depth $q_e = 5$, $p_c = 0.01$

Figure 1. Useful Region for $p_c = 0.001$ and $p_c = 0.01$

B. Finite Length Performance of Min-Sum based decoders

In this section we evaluate the finite-length performance of three noisy Min-Sum based decoders: the Min-Sum (MS), the Offset Min-Sum (OMS) [15] and the Self-Corrected Min-Sum (SCMS) [16]. The objective is to determine if a correction circuit "plugged into" the noisy MS decoder can improve the robustness of the decoder to hardware noise. The characteristic of the SCMS decoder is to *erase* (*i.e.* set to zero) any variable-to-check message that changed its sign with respect to the previous iteration. However, the same message cannot be erased during two consecutive iterations. Therefore, binary values $E_{m,n} \in \{0, 1\}$ are used to record whether or not the corresponding variable-to-check message has been erased at the previous iteration. These values are all initialized to zero (meaning: "message **not** erased at previous iteration"). The noisy SCMS decoder performs the same computations as the noisy MS (Section III-D), except that the VN processing step further includes a *correction step*, as follows:

- VN-processing: for $\forall\ n \in \{1, \ldots, N\}$ and $m \in \mathcal{H}(n)$

$$\alpha_{m,n}^{(\ell)} = \mathbf{a}_{\text{pr}}\left(\tilde{\gamma}_n^{(\ell)}, -\beta_{m,n}^{(\ell)}\right);$$
if $E_{m,n} = 0$ and $\text{sgn}\left(\alpha_{m,n}^{(\ell)}\right) \neq \text{sgn}\left(\alpha_{m,n}^{(\ell-1)}\right)$
$\alpha_{m,n}^{(\ell)} = 0\,;\ E_{m,n} = 1\,;$ // message erased
else $E_{m,n} = 0\,;$ // message not erased
end

The body enclosed between the **if** condition and the matching **end** is referred to as correction step. In practical implementation, only the sign of previous variable-to-check messages needs to be stored. Note that the correction step is implemented with reliable circuitry, which can be reasonably assumed, since the required circuitry is very simple (see also the discussion concerning the sign of check-to-variable messages in Section III-D). Alternatively, one can think of the correction circuit as a *noiseless patch* applied to the noisy MS decoder, in order to improve its robustness to hardware noise.

Simulation results presented below were obtained for the $(N = 1008, K = 504)$, $(d_v = 3, d_c = 6)$-regular LDPC code, available online at [17]. Fig. 2 shows the bit error rate performance of the three decoders. The colors blue, green and red are used respectively for MS, OMS, and SCMS decoders.

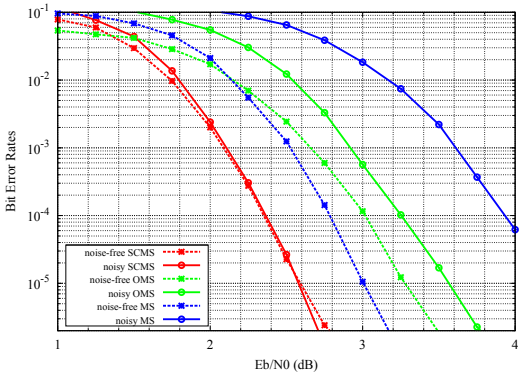


Figure 2. BER for $q_e = 4$, $p_a = 0.01$ and $p_c = 0.01$

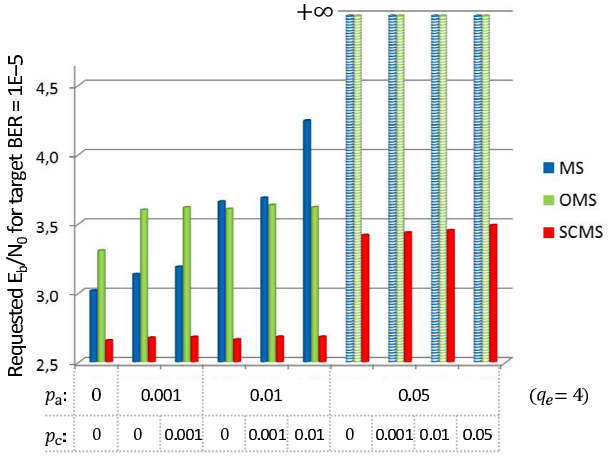


Figure 3. Eb/N0 required for a BER=10^{-5}.

The solid lines are used for the performance of the noisy decoders, while the dotted lines are used to represent the noiseless decoder performance. Noisy decoders have probabilistic components with parameters $q_e = 4$, $p_a = 0.01$, and $p_c = 0.01$. It is worth noting that the probability of hardware-induced errors on the sign of the a posteriori information is less than 10^{-6}, which explains the absence of an error floor above this value.

Although the OMS decoder usually outperforms the MS decoder, it is not the case here for noiseless decoders. Because check-node messages are quantized on only $q = 4$ bits, the smallest offset factor $\delta = 1$ is still too high to improve the performance of the MS decoder. However for noisy hardware, the OMS decoder is more robust and provides better performance than the MS decoder. The most impressive performance is that of the noisy SCMS decoder, which exhibits almost the same performance as the noiseless SCMS. The explanation is to be sought in the intrinsic property of the SCMS to detect unreliable messages and discard them from the decoding process.

With the objective of having a better understanding of the impact of hardware noise on the three decoders, additional simulations have been carried out with various noise parameters. Fig. 3 shows the Eb/N0 values in dB corresponding to a target BER $= 10^{-5}$ for $q_e = 4$ and various (p_a, p_c) parameters. The results show that the MS decoder is very sensitive to hardware noise as the value of Eb/N0 increases drastically with p_a and p_c. At low level of noise ($p_a \leq 0.001$) the OMS decoder needs higher EbN0 to provide the same error probability than the MS decoder, but it outperforms the MS decoder as the hardware noise increases. However, none of these decoders can achieve a target BER $= 10^{-5}$ for $p_a = 0.05$ (this is illustrated in the figure by an infinite EbN0 value). Fig. 3 also confirms the excellence performance of the SCMS on noisy hardware, since it exhibits performance similar to the noiseless SCMS decoder when $p_a \leq 0.01$. It can also achieve the target BER value when $p_a = 0.05$, with a performance penalty of 0.8 dB.

VI. Conclusion

In this paper we investigated the performance of MS-based decoders on noisy hardware. We derived density evolution equations for the noisy MS decoder, and analyzed the decoder in terms of useful regions and target-BER thresholds. We further evaluated the finite length performance of several MS-based decoders, for various parameters of the hardware noise models. We highlighted the excellent performance of the SCMS decoder, which is due to its intrinsic ability to detect and discard unreliable messages during the iterative decoding process. Finally, the results of our work may serve as guidelines for the design of noisy arithmetic components for Min-Sum-based decoders.

Acknowledgment

This work was supported by the Seventh Framework Programme of the European Union, under Grant Agreement number 309129 (i-RISC project).

References

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

[2] ——, "Reliable computation in computing systems designed from unreliable components," *Bell System Technical Journal*, vol. 47, pp. 2339–2366, 1968.

[3] A. V. Kuznetsov, "Information storage in a memory assembled from unreliable components," *Problemy Peredachi Informatsii*, vol. 9, no. 3, pp. 100–114, 1973.

[4] S. K. Chilappagari, M. Ivkovic, and B. Vasic, "Analysis of one step majority logic decoders constructed from faulty gates," in *Proc. of IEEE Int. Symp. on Information Theory*, 2006, pp. 469–473.

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

[6] C. Winstead and S. Howard, "A probabilistic LDPC-coded fault compensation technique for reliable nanoscale computing," *IEEE Trans. on Circuits and Systems II: Express Briefs*, vol. 56, no. 6, pp. 484–488, 2009.

[7] Y. Tang, C. Winstead, E. Boutillon, C. Jego, and M. Jezequel, "An ldpc decoding method for fault-tolerant digital logic," in *IEEE Int. Symp. on Circuits and Systems (ISCAS)*, 2012, pp. 3025–3028.

[8] A. M. Hussien, M. S. Khairy, A. Khajeh, A. M. Eltawil, and F. J. Kurdahi, "A class of low power error compensation iterative decoders," in *IEEE Global Telecom. Conf. (GLOBECOM)*, 2011, pp. 1–6.

[9] L. R. Varshney, "Performance of LDPC codes under faulty iterative decoding," *IEEE Trans. on Inf. Theory*, vol. 57, no. 7, pp. 4427–4444, 2011.

[10] S. Yazdi, H. Cho, Y. Sun, S. Mitra, and L. Dolecek, "Probabilistic analysis of gallager b faulty decoder," in *IEEE Int. Conf. on Communications (ICC)*, 2012, pp. 7019–7023.

[11] S. Yazdi, C. Huang, and L. Dolecek, "Optimal design of a gallager b noisy decoder for irregular ldpc codes," *IEEE Comm. Letters*, vol. 16, no. 12, pp. 2052–2055, 2012.

[12] S. Yazdi, H. Cho, and L. Dolecek, "Gallager b decoder on noisy hardware," *IEEE Trans. on Comm.*, 2013.

[13] R. G. Gallager, "Low density parity check codes," MIT Press, Cambridge, 1963, research Monograph series.

[14] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. on Inf. Theory*, vol. 27, no. 5, pp. 533–547, 1981.

[15] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X. Hu, "Reduced-complexity decoding of ldpc codes," *IEEE Trans. on Communications*, vol. 53, no. 8, pp. 1288–1299, 2005.

[16] V. Savin, "Self-corrected min-sum decoding of LDPC codes," in *Proc. of IEEE Int. Symp. on Information Theory (ISIT)*, 2008, pp. 146–150.

[17] D. J. C. MacKay. Encyclopedia of sparse graph codes. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html