Energy Optimization of Quantized Min-Sum Decoders for Protograph-Based LDPC Codes

Mohamed Yaoumi[1], Elsa Dupraz[1], François Leduc-Primeau[2], Frederic Guilloud[1]

[1] IMT Atlantique, Lab-STICC, UMR CNRS 6285, F-29238, France.

[2] Department of Electrical Engineering, Polytechnique Montreal, QC, Canada.

Abstract

This paper considers protograph-based LDPC codes, and proposes an optimization method to select protographs that minimize the energy consumption of quantized Min-Sum decoders. This method first estimates the average number of iterations required by the decoder, and includes this estimate into two high level models that evaluate the decoder energy consumption. The optimization problem is then formulated as minimizing the energy consumption of the decoder while satisfying a performance criterion on the Frame Error Rate. Finally, an optimization algorithm based on differential evolution is introduced. Protograph optimized for energy consumption shows a gain in energy of approximately 15% compared to a baseline protograph optimized for performance only.

I. Introduction

Reducing the energy consumption of telecommunication systems may allow to improve the communication capabilities of systems with limited resources. This energy consumption comes for a large part from the transmission power of the emitter, but the processing power at the receiver is also non-negligible for short-length communications [1]. In addition, error-correction decoders are known to use a large part of this processing power. Therefore, the objective of this paper is to reduce the energy consumption of the error-correction part. It considers Low Density Parity Check (LDPC) codes as a particular family of error-correction codes which were retained in the 5G standardization process.

The problem of reducing the energy consumption of LDPC decoders has received increased attention recently. In [2], the energy consumption of hard-decision LDPC decoders is estimated from the number of computation operations realized in the decoder, and from the average length of wires in the decoding circuit. Then, [3] considers more powerful LDPC decoders working on discrete message alphabets, and proposes a method to reduce the message alphabet size. This allows to lower both memory and wire energy consumption in the decoder. Alternatively, [4]–[6] propose to optimize the code parameters in order to reduce the decoder energy consumption. For instance, [4], [6] optimize the code degree distribution in order to minimize the decoder complexity. However, [6] considers hard-decision Gallager B decoders with poor performance, and [4] considers infinite precision sum-product decoding algorithms which cannot be implemented directly on hardware. To circumvent these issues, in [5], we proposed two models to evaluate the energy consumption of a more practical quantized Min-Sum decoder [7]. The first model evaluates the energy consumption from the number of operations realized in the decoder, and the second model counts the number of memory writes in the decoder.

In this paper, we also study the quantized Min-sum decoder of [7], and consider Quasy-Cyclic (QC) codes constructed from protographs, for their easy hardware implementation. We rely on the two energy models introduced in [5]. Since these two models depend on the average number of iterations performed in the decoder, we introduce a method to precisely evaluate this number, depending on the codeword length and on the considered protograph. This method relies on the finite-length performance analysis based on Density Evolution (DE) initially described in [8]. Based on this evaluation, we then introduce a novel optimization framework to minimize the energy consumption of the decoder, while satisfying a certain performance criterion on the Frame Error Rate (FER). We start by showing that minimizing the decoder energy consumption can be equivalently restated as minimizing the product of a function of the average code degrees with the sum of FERs at successive decoder iterations. Interestingly, this shows that minimizing the decoder energy consumption requires to improve the decoder performance. We then develop an optimization algorithm that relies on a modified Differential Evolution algorithm [9] in order to select protographs which minimize the decoder energy consumption. Protograph optimized for energy consumption shows a gain in energy of about 15% compared to a baseline protograph optimized for performance only.

The outline of the paper is as follows. Section II introduces our notation and assumptions for LDPC codes and decoders. Section III presents our method to estimate the average number of iterations required by the decoder. Section IV describes the two energy models we consider. Section V introduces our optimization method. Section VI shows simulation results.

II. LDPC CODES

Throughout the paper, we use $[\![1, K]\!]$ to denote the set of integers with values between 1 and K, and we use $|\mathcal{S}|$ to denote the cardinal of the set $\mathcal{S}$. We consider the transmission of a codeword $\mathbf{x}$ of length N over an Additive White Gaussian Noise (AWGN) channel of variance σ^2 with Binary-Shift-Keying (BPSK) modulation. We represent the channel output by a vector $\mathbf{y}$ of length N. We also define the channel Signal-to-Noise (SNR) ratio as $\xi = 1/\sigma^2$.

A. LDPC codes construction

An LDPC code of rate R is defined by its binary parity check matrix H of size $M \times N$. The matrix H can be represented by a Tanner graph which connects N Variable Nodes (VN) v_i ($i \in [\![1, N]\!]$) to M Check Nodes (CN) c_j ($j \in [\![1, M]\!]$). In the Tanner graph, there is an edge between VN v_i and CN c_j if the component $H_{i,j}$ of H is equal to 1. The set of CNs connected to VN v_i is denoted $\mathcal{C}_i$, and the set of VNs connected to CN c_j is denoted $\mathcal{V}_j$. The degree of VN v_i is denoted $d_{v_i} = |\mathcal{C}_i|$, and the degree of CN c_j is denoted $d_{c_j} = |\mathcal{V}_j|$.

In this paper, we consider Quasi-Cyclic (QC) LDPC codes constructed from protographs for their good performance and easy hardware implementation. A protograph can be represented by a small matrix S of size $m \times n$. In order to construct a QC LDPC code from a protograph, we consider the two step lifting procedure described in [10], which allows to reduce the amount of cycles in the parity check matrix H. In this procedure, the protograph S is first copied Z_1 times, where Z_1 is the first lifting factor. The edges of the obtained matrix are then interleaved in order to produce a base matrix B of size $Z_1 m \times Z_1 n$. Secondly, all the base matrix coefficients are replaced by circulant matrices of size $Z_2 \times Z_2$, where Z_2 is the second lifting factor. The final code performance depends on several factors, including the choice of the original protograph, the lifting factors Z_1 and Z_2, and the circulant matrices.

B. Quantized Min-Sum decoder

In this paper, we consider the Min-Sum decoder implementation proposed in [7], as this implementation allows for high degree of parallelism and reduced decoding latency. In the decoder, messages are quantized on q bits and take values between $-Q$ and $+Q$, where $Q = 2^{q-1} - 1$. The quantization step is denoted by s, and the quantization function is given by

$$\mathcal{Q}(x) = \operatorname{sgn}(x) \min\left(s \left\lfloor \frac{|x|}{s} + \frac{1}{2} \right\rfloor, Q \right),$$

where sgn is the sign operator. In the quantization function $\mathcal{Q}$, the $\min$ operator realizes message saturation.

In the decoder, messages are initialized at each VN v_i, $i \in [\![1, N]\!]$, from the quantized Log-Likelihood Ratio (LLR) values

$$r_i = \mathcal{Q}\left(\alpha \log\left(\frac{\Pr(x_i = 1|y_i)}{\Pr(x_i = -1|y_i)}\right)\right) = \mathcal{Q}\left(\alpha \frac{2y_i}{\sigma^2}\right), \tag{1}$$

where y_i is the i-th channel output and α is a scaling parameter. Then, the decoder works in a maximum of L iterations. At iteration $\ell \in [\![1, L]\!]$, VN messages are calculated as the sum of incoming messages

$$\mu_i^{(\ell)} = r_i + \sum_{j \in \mathcal{C}_i} \gamma_{j \to i}^{(\ell-1)}, \tag{2}$$

where $\gamma_{j \to i}^{(\ell-1)}$ denotes message from CN c_j to VN v_i. The sum values $\mu_i^{(\ell)}$ are stored on $q + q_s$ bits, where the value of q_s is set as $q_s = \lceil \log_2(\max_{i \in [\![1,n]\!]} (d_{v_i}) + 1) \rceil$, in order to avoid message saturation. However, when sent to CNs, the values $\mu_i^{(\ell)}$ are reduced to q bits.

Next, messages $\gamma_{j \to i}^{(\ell)}$ from CN c_j to VN v_i are calculated in two steps as

$$\mu_{i \to j}^{(\ell)} = \mu_i^{(\ell)} - \gamma_{j \to i}^{(\ell-1)} \tag{3}$$

$$\gamma_{j \to i}^{(\ell)} = \left(\prod_{i' \in \mathcal{V}_j \setminus i} \mathrm{sgn}\left(\mu_{i' \to j}^{(\ell-1)}\right)\right) \times \max\left(\min_{i' \in \mathcal{V}_j \setminus i} \left|\mu_{i' \to j}^{(\ell-1)}\right| - \lambda, 0\right). \tag{4}$$

where λ is the offset parameter. The decoder stops if the maximum number of iterations L is reached, or whenever a stopping criterion is satisfied. As stopping criterion, we verify all the parity check equations defined by the matrix H are verified.

Note that in the architecture of [7], a VN message μ_i is calculated as the sum of all incoming messages, while in standard decoders, the message from the current edge is excluded from the sum. In this architecture, the message $\gamma_{j \to i}^{(\ell-1)}$ from the current edge is subtracted during CN processing, see (3). In addition, the implementation of [7] considers the layered serial-C scheduling described in [11]. In this scheduling, a VN is updated each time one of its neighborhing CNs is updated. And CNs are updated serially in successive layers, where a layer is composed by Z_2 CNs which are updated in parallel.

III. Finite-length performance evaluation of the decoder

DE [12] is a common tool to evaluate the performance of an LDPC decoder. For a given protograph, DE allows to predict the error probability of the decoder under the assumption that the codeword length is infinite. In our case, this error probability can be evaluated by considering multi-edge type DE [12] for a quantized decoder. However, since the energy consumption may depend on some finite-length effect, we here consider the method proposed in [8], which uses DE in order to predict the finite-length performance of the decoder. In this method, the FER performance is evaluated for a codeword of length N, at SNR value ξ, and iteration ℓ, from the following formula:

$$B_N^{(\ell)}(\xi) = \int_0^{\frac{1}{2}} B_\infty^{(\ell)}(x)\, \Phi_{\mathcal{N}}\left(x; p_0, \frac{p_0(1-p_0)}{N}\right) dx. \tag{5}$$

In this expression, $p_0 = \frac{1}{2}\left(1 - \text{erf}\left(R \times \xi\right)\right)$, and erf is the error function of the Gaussian distribution. In addition, $\Phi_{\mathcal{N}}(x; \mu, v^2)$ represents the density function at x of a Gaussian distribution with mean μ and variance v^2. Finally, $B_\infty^{(\ell)}(x) = 1 - (1 - P_\infty^{(\ell)}(x))^N$, and $P_\infty^{(\ell)}(x)$ gives the decoder error probability calculated by standard DE for variance value $v^2 = \frac{1}{2\left(\text{erf}^{-1}(1-2x)\right)^2}$.

In order to evaluate its energy consumption, we also need to predict the number of iterations required by the decoder at a given length N. To do so, we now describe how we propose to evaluate the average number of iterations $L_N(\xi)$ at length N. For this, we use $L_{N,\xi}(j)$ to denote the number of iterations needed to decode a given frame $j \in [\![1, 2^K]\!]$, where $K = N - M$, depending on the codeword length N and on the SNR ξ. The average number of iterations $L_N(\xi)$ can then be expressed as

$$L_N(\xi) = E\left[L_{N,\xi}^{(\ell)}\right] = \sum_{j=1}^{2^K} \frac{1}{2^K} L_{N,\xi}(j) = \sum_{j=1}^{2^K} \frac{1}{2^K} \sum_{\ell=1}^{L} \mathbb{1}_{\mathcal{A}_{N,\xi}^{(\ell-1)}}(j) \tag{6}$$

where $\mathcal{A}_{N,\xi}^{(\ell-1)}$ is the set of frames which do not satisfy the stopping criterion at $\ell - 1$ in the decoder, and $\mathbb{1}$ is the indicator function. We then denote by $\mathcal{B}_{N,\xi}^{(\ell-1)}$ the set of frames imperfectly decoded at iteration $\ell - 1$. We assume that the stopping criterion considered in the decoder is perfect, which gives that $\mathcal{B}_{N,\xi}^{(\ell-1)} = \mathcal{A}_{N,\xi}^{(\ell-1)}$. Under this assumption, we have that

$$L_N(\xi) = \sum_{j=1}^{2^K} \frac{1}{2^K} \sum_{\ell=1}^{L} \mathbb{1}_{\mathcal{B}_{N,\xi}^{(\ell-1)}}(j) \tag{7}$$

$$= \sum_{\ell=1}^{L} B_N^{(\ell-1)}(\xi) \tag{8}$$

where $B_N^{(\ell-1)}(\xi)$ is defined in (5), and (8) is obtained by permuting the sums in (7).

The above derivation relies on the condition that $\mathcal{B}_{N,\xi}^{(\ell-1)} = \mathcal{A}_{N,\xi}^{(\ell-1)}$. Indeed, in order to determine if it should stop at iteration $\ell - 1$, the decoder relies on a stopping criterion. However, for some frames j, this stopping criterion may be verified while the frame is not correctly decoded, thus leading to $j \in \mathcal{B}_{N,\xi}^{(\ell-1)}$ but $j \notin \mathcal{A}_{N,\xi}^{(\ell-1)}$. The probability of this event depends on the code cycle distribution and minimum distance. In what follows, since we consider long codewords, we choose to neglect this event and evaluate the average number of iterations as (8). The accuracy of (8) will be evaluated from numerical simulations later in the paper.

The FER (5) and the average number of iterations (8) are evaluated by taking into account channel uncertainty at length N. However, these evaluations do not take into account code cycles which can also degrade the decoder performance at finite length. These methods are therefore well-suited to evaluate the code performance for codewords of moderate to long sizes, starting from around 5000 bits.

IV. Energy Models

In this section, we describe two models that evaluate the energy consumption of the quantized Min-Sum decoder. The first model counts the number of memory writes in the decoder (memory model), and the second model counts the number of operations realized by the decoder (complexity model). These two models were first given in [5], and we update them with the evaluation of the average number of iterations introduced in Section III. In order to express these models, we define $\tilde{d}_v = \frac{1}{n}\sum_{i=1}^{n} d_{v_i}$ and $\tilde{d}_c = \frac{1}{m}\sum_{i=1}^{n} d_{c_j}$ as the average VN and CN degrees, respectively, with $\tilde{d}_v(1-R) = \tilde{d}_c$.

A. Memory model

The memory model evaluates the number of memory writes in the decoder. At each iteration, a CN of degree d_c stores one 1 sign bit for each of the d_c messages, and uses $2(q-1)$ bits to store the 2 minimum values among incoming messages. This results in a total of $d_c + 2(q-1)$ bits. Then, at each iteration, a VN of degree d_v only stores the sum (2) of incoming messages, which requires $q + q_s$ bits.

Therefore, at SNR ξ, the memory energy model is given by

$$E_m(\xi) = L_N(\xi) N E_{\text{bit}} \left((q+q_s)\tilde{d}_v + (1-R)(\tilde{d}_c + 2(q-1)) \right) \tag{9}$$

where E_{bit} is the energy needed to write one bit in memory. We see that this energy model depends on the considered protograph through the values of $\tilde{d}_v$ and $\tilde{d}_c$, and also through the number of iterations $L_N(\xi)$. It also depends on the codeword length N, on the code rate R, and on the number of quantization bits q.

B. Complexity model

The complexity model evaluates the number of computation operations realized by the decoder. Due to the serial-C scheduling, a VN is updated each time one of its neighborhing CNs is updated. Once a VN is updated, the sum (3) is first computed. Then, the sum of incoming messages (2) is updated by calculating $\mu_i^{(\ell)} = \mu_{i\to j}^{(\ell)} + \gamma_{j\to i}^{(\ell)}$. As a result, VN update requires $2d_v$ sums on $q+q_s$ bits. In addition, during a CN update, the sign of each message is first computed from a XOR operation on $2d_c - 1$ bits. The two minimum values are calculated from merge-sort circuits [13] which require $\lfloor \frac{d_c}{2} \rfloor + 2(\lceil \frac{d_c}{2} \rceil - 1)$ comparisons, and all the comparisons are performed on inputs of $q-1$ bits. The above number of comparisons depends on whether the value of d_c is odd or even. Here, in order to simplify the expressions, we consider the worst case, that is when d_c is odd. In this case, the number of required comparisons is given by $\frac{3}{2}(d_c-1)$. Then, applying the offset in (4) requires one substraction and one comparison, on inputs of q bits. Finally, verifying the stopping criterion demands d_c XOR operations per VN.

As a result, at SNR ξ, the complexity energy model is given by

$$\begin{aligned} E_c(\xi) = {} & L_N(\xi) N \Big(E_{\text{add}} (2(q+q_s)\tilde{d}_v + \frac{3}{2}(q-1)(\tilde{d}_c - 1) + 2) \Big) \\ & + L_N(\xi) N \Big(E_{\text{xor}} (2\tilde{d}_c + \tilde{d}_v - 1) \Big) \end{aligned} \tag{10}$$

where we assumed that a comparison has the same complexity as an addition, and E_{add} and E_{xor} represent the elementary energy consumption of 1-bit addition and XOR-2 operation, respectively.

V. Energy Optimization Method

We now propose a method to optimize the decoder energy consumption, based on the above two energy models. We first formulate the optimization problem, and then propose a simplified equivalent formulation.

A. *Optimization problem*

We want to optimize the energy consumption of the decoder while satisfying a certain performance criterion. The performance criterion is defined as a condition on the maximum FER $B_{\max}$ that can be tolerated at SNR value ξ. Here, we want to select the best protograph S, and therefore assume that almost all the other code and decoder parameters are fixed (rate R, codeword length N, number of quantization bits q, maximum number of iterations L). As a result, the optimization problem can be expressed as

$$\min_{S} E(\xi) \quad \text{s.t.} \quad B_N^{(L)}(\xi) < B_{\max}, \tag{11}$$

where $B_N^{(L)}(\xi)$ is the FER estimated at length N and defined in (5), and the energy term $E(\xi)$ may represent either the memory energy model, or the complexity energy model, or a weighted combination of both. In the above formulation, the terms $E(\xi)$ and $B_N^{(L)}(\xi)$ are calculated for parameters $\overline{\lambda}$ and $\overline{\alpha}$ such that

$$\overline{\lambda}, \overline{\alpha} = \arg\min_{\lambda,\alpha} B_N^{(L)}(\xi) \tag{12}$$

where α is the scaling parameter introduced in (1) and lambda is the offset parameter introducted in (4). Note that the optimal values of α and λ strongly depend on the considered protograph.

The FER $B_N^{(L)}(\xi)$ and the average number of iterations $L_N(\xi)$ do not have explicit analytical expressions, although they can be evaluated numerically. Therefore, a common solution for protograph optimization [14] consists of adapting a genetic algorithm called Differential Evolution [9] and initially proposed for continuous optimization. However, while previous works [14] consider protograph optimization for performance only, the main difficulty in our case is to address the tradeoff between energy consumption and performance. In particular, Differential Evolution is not well-suited for constrained optimization, as the algorithm should be initialized inside the feasible set, and in our case, the feasible set is the set of protographs that satisfy the constraint $B_N^{(L)}(\xi) < B_{\max}$. In order to soften this issue, we now derive an equivalent optimization problem that shows that the decoder energy consumption is directly related to its performance.

B. Equivalent optimization problem

For the two considered energy models, given that the parameters R, N, q are fixed, the optimization problem (11) can be restated as:

$$\min_{S} \left(f(\tilde{d}_v) L_N(\xi) \right) \quad \text{s.t.} \quad B_N^{(L)}(\xi) < B_{\max}, \tag{13}$$

where the expression of the function f depends on the considerd energy model. In addition, by replacing $L_N(\xi)$ by its expression in (8), we further obtain that the optimization problem (11) is equivalent to

$$\min_{S} \left(f(\tilde{d}_v) \sum_{\ell=1}^{L} B_N^{(\ell-1)}(\xi) \right) \quad \text{s.t.} \quad B_N^{(L)}(\xi) < B_{\max} \tag{14}$$

This formulation shows that the two key criterion for energy optimization are the average VN degree $\tilde{d}_v$ and the FER performance at successive iterations. In the next section, we use the equivalent formulation (14) to propose an efficient optimization algorithm based on Differential Evolution.

C. Optimization method

Differential Evolution [9] is a genetic algorithm which starts from an initial population of K protographs. The initial protographs are generated at random under the constraint that the maximum protograph coefficient is given by $d_{\max}$, and that VN and CN degrees are greater than 2. This second constraint avoids generating protographs with poor minimum distance. Then, the Differential Evolution algorithm works over T iterations. At each iteration, the algorithm generates K new candidate protographs according to recombination operations described in [9]. During recombination operations, the obtained non-integer protograph coefficients are rounded to the closest integer value, and they are also saturated in order to keep coefficients between 0 and $d_{\max}$. Next, the algorithm compares the criterion value of the newly generated protograph $S^{(k,t)}$, for $k \in [\![1, K]\!]$, with the criterion value of the previous protograph $S^{(k,t-1)}$. If the criterion for $S^{(k,t)}$ is best, then this protograph is included in the population. Otherwise, the previous protograph $S^{(k-1,t)}$ is conserved in the population. After T iterations, the protograph with the best criterion among the population is chosen as the solution to the optimization problem. Although this algorithm does not give guarantees on the quality of the retained solution, is ensures that the criterion is reduced from iteration to iteration.

TABLE I

INFINITE-LENGTH THRESHOLDS AND FINITE-LENGTH ENERGY VALUES OF THE PROTOGRAPHS FOR $\xi = 1.45$dB AND $N = 10000$.

Protograph	Threshold	E_c	E_m
$S_{\text{opt}} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 0 & 4 & 1 \end{bmatrix}$	1.18 dB	35.9 nJ	508 nJ
$S_0 = \begin{bmatrix} 2 & 1 & 3 & 2 \\ 5 & 1 & 1 & 0 \end{bmatrix}$	1.15 dB	46.2 nJ	733 nJ
$S_c = \begin{bmatrix} 0 & 1 & 2 & 5 \\ 2 & 2 & 0 & 2 \end{bmatrix}$	1.20 dB	38.8 nJ	585 nJ
$S_m = \begin{bmatrix} 3 & 2 & 1 & 2 \\ 0 & 1 & 1 & 4 \end{bmatrix}$	1.21 dB	38.8 nJ	585 nJ

In order to apply this method to our problem, we start from the optimization problem (14). As a criterion for protograph selection at each iteration, we consider the function g defined as

$$g(S) = \left(f(\tilde{d}_v) \sum_{\ell=1}^{L} B_N^{(\ell-1)}(\xi) \right), \tag{15}$$

In addition, during the T iterations of the algorithm, we consider a relaxed constraint on the performance, that is that the protograph threshold obtained from standard DE must be larger than a certain value $\xi^\star$ in order for this protograph to be included in the population. This relaxed constraint allows to increase the feasible set when runing Differential Evolution. Finally, after the T iterations, we select the protograph that both minimizes the criterion g and satisfies the FER constraint $B_{\max}$.

VI. SIMULATION RESULTS

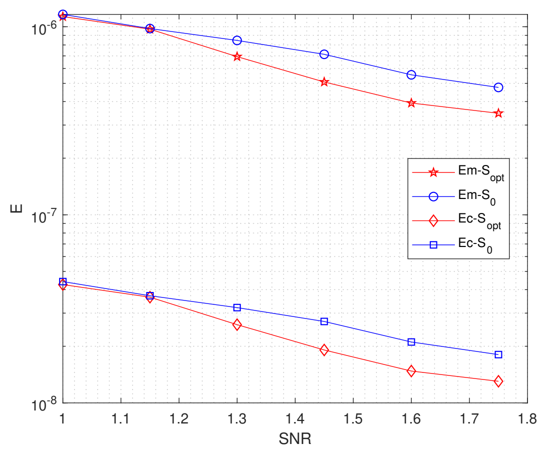
We now provide results for the optimization method described in Section V. As a baseline, we consider the protograph S_0 given in Table I. This protograph was optimized for performance only, by minimizing the asymptotic threshold provided by standard density evolution.

For our optimization, we considered protographs of size $m = 2$, $n = 4$, with maximum protograph coefficient $d_{\max} = 6$. The code parameters were set as $N = 10000$, $R = 0.5$, and the decoder parameters are $q = 6$, $L = 50$. As relaxed performance constraint, we considered $\xi^* = 1.25$dB. This value was chosen in order to take some margin compared to the protograph

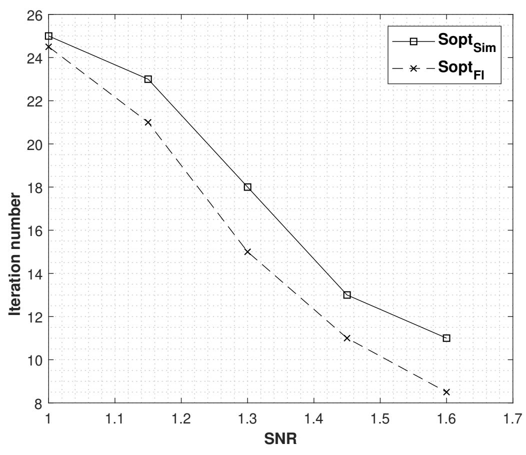
S_0 which was optimized for performance only and which has a threshold value of 1.15dB. In addition, as final performance constraint, we considered $B_{\text{max}} = 10^{-2}$ at value $\xi = 1.45$dB. From these parameters, we applied our optimization algorithm for a population of $K = 112$ protographs, over $T = 12$ iterations, for the two energy models. The optimization method returned the protograph S_{opt} given in Table I, for the two models. This protograph has FER $B_N^{(L)}(\xi) = 0.0049$ at $\xi = 1.45$dB. In Table I, we also give the protographs S_m and S_c which were obtained in [5]. We now compare the performance of these three protographs in terms of energy and performance.

In order to compare the energy values, for illustrative purposes, we substituted the energy constants with rough estimates. Based on the estimate for a 32-bit addition reported in [15], we set $E_{\text{add}} = 3.13$ fJ, and since a 1-bit adder contains two XOR-2 gates, $E_{\text{xor}} = 1.56$ fJ. We base the energy of writing one bit in memory on the estimate of 10 pJ for a 64-bit access from an 8KB cache, yielding an average of $E_{\text{bit}} = 0.156$ pJ. The energy values for the two models and for each protograph are shown in Table I. We observe that S_0 shows a high energy consumption, which is due to the fact that this protograph was optimized for performance only. On the opposite, S_{opt} has the lowest energy consumption, with a gain of approximately 15% compared to S_0. To verify these results, Figure 1 (a) shows the energy values with respect to SNR for the two energy models, for protographs S_{opt} and S_0. The figure confirms that protograph S_{opt} has lower energy consumption than S_0 at any considered SNR.

In terms of performance, Figure 1 (b) first compares for S_{opt} the average number of iterations estimated from the finite-length method of Section III, with the average number of iterations measured from Monte Carlo simulations. This shows that the proposed method estimates the average number of iterations with a difference of approximately 5 iterations which probably comes from imperfect stopping criterion and from short code cycles. Then, Figure 2 compares the BER and FER values for S_0 and S_{opt}, obtained from both the finite-length method of Section III and from Mont Carlo simulations. We first observe that the method proposed in Section III provides a good approximation of the decoder performance. Second, we see that although S_0 has a better performance in the waterfall, it shows an important error floor at higher SNR values, which can be observed from the FER curve.

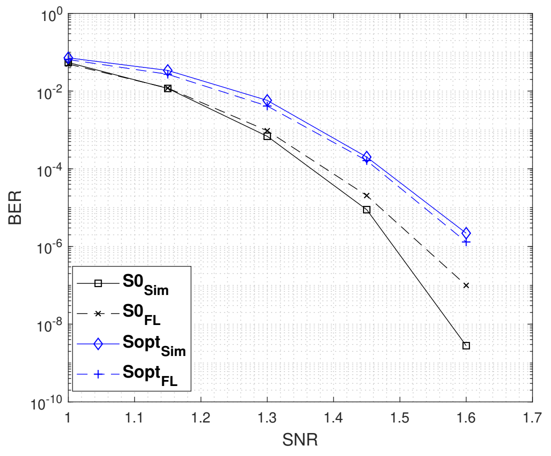


(a)

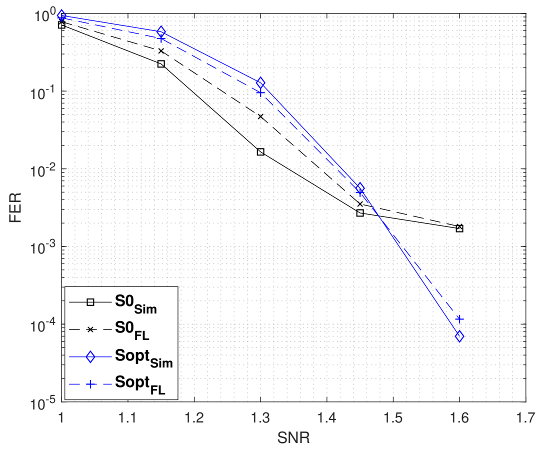


(b)

Fig. 1. (a) Energy with respect to SNR for S_{opt} and S_0 for the two energy models, (b) Average number of iterations with respect to SNR for S_{opt} estimated using the finite length method and the Monte Carlo simulation.



(a)



(b)

Fig. 2. Performance comparison between codes constructed from protographs S_0 and S_{opt}. The FER performance was evaluated from finite-length DE (FL) and from Monte Carlo simulations (sim) (a) BER vs SNR performance (b) FER vs SNR performance

VII. Conclusion

In this paper, we introduced an optimization framework to select protographs that minimize the energy consumption of quantized Min-Sum LDPC decoders, while satisfying a performance criterion expressed in termes of FER. The proposed optimization method was based on the estimation of the average number of iterations required by the decoder, and on a Differential

Evolution algorithm to optimize the protographs. Future works will be dedicated to optimizing not only the protograph, but also other code and decoder parameters.

VIII. Acknowledgements

This work was supported by the grant ANR-17-CE40-0020 of the French National Research Agency ANR (project EF-FECtive).

References

[1] K. Ganesan, P. Grover, J. Rabaey, and A. Goldsmith, “On the total power capacity of regular-LDPC codes with iterative message-passing decoders,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 2, pp. 375–396, 2016.

[2] ——, “Towards approaching total-power-capacity: Transmit and decoding power minimization for LDPC codes,” *arXiv preprint arXiv: 1504.01019, 2015*, 2015.

[3] T. T. Nguyen-Ly, K. Le, V. Savin, D. Declercq, F. Ghaffari, and O. Boncalo, “Non-surjective finite alphabet iterative decoders,” in *IEEE Int. Conf. on Communications (ICC)*, 2016, pp. 1–6.

[4] B. Smith, M. Ardakani, W. Yu, and F. R. Kschischang, “Design of irregular LDPC codes with optimized performance-complexity tradeoff,” *IEEE Trans. on Communications*, vol. 58, no. 2, 2010.

[5] M. Yaoumi, F. Leduc-Primeau, E. Dupraz, and F. Guilloud, “Optimization of protograph ldpc codes based on high-level energy models,” in *2019 16th International Symposium on Wireless Communication Systems (ISWCS)*. IEEE, 2019, pp. 6–10.

[6] W. Yu, M. Ardakani, B. Smith, and F. Kschischang, “Complexity-optimized low-density parity-check codes for gallager decoding algorithm B,” in *2005 Int. Symp. on Information Theory (ISIT)*, 2005, pp. 1488–1492.

[7] E. Dupraz, F. Leduc-Primeau, and F. Gagnon, “Low-latency LDPC decoding achieved by code and architecture co-design,” in *2018 IEEE 10th International Symposium on Turbo Codes & Iterative Information Processing (ISTC)*. IEEE, 2018, pp. 1–5.

[8] F. Leduc-Primeau and W. J. Gross, “Finite-length quasi-synchronous LDPC decoders,” in *9th Int. Symp. on Turbo Codes and Iterative Information Processing (ISTC)*, 2016, pp. 325–329.

[9] R. Storn and K. Price, “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[10] D. G. Mitchell, R. Smarandache, and D. J. Costello, “Quasi-cyclic LDPC codes based on pre-lifted protographs,” *IEEE Trans. on Information Theory*, vol. 60, no. 10, pp. 5856–5874, 2014.

[11] E. Sharon, S. Litsyn, and J. Goldberger, “Efficient serial message-passing schedules for LDPC decoding,” *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4076–4091, 2007.

[12] T. Richardson, R. Urbanke *et al.*, “Multi-edge type LDPC codes,” in *Workshop honoring Prof. Bob McEliece on his 60th birthday, California Institute of Technology, Pasadena, California*, 2002, pp. 24–25.

[13] F. Leduc-Primeau, F. R. Kschischang, and W. J. Gross, “Modeling and energy optimization of LDPC decoder circuits with timing violations,” *IEEE Trans. on Commun.*, vol. 66, no. 3, pp. 932–946, 2018.

[14] F. Ye, E. Dupraz, Z. Mheich, and K. Amis, “Optimized rate-adaptive protograph-based ldpc codes for source coding with side information,” *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 3879–3889, 2019.

[15] M. Horowitz, “1.1 computing’s energy problem (and what we can do about it),” in *2014 IEEE Int. Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 10–14.