Analysis of LDPC code syndrome entropy based on subgraphs

David Matas, Meritxell Lamarca
Universitat Politècnica de Catalunya, Barcelona, Spain, Email: (david.matas, meritxell.lamarca)@upc.edu

***Abstract*—We propose a method to bound the syndrome entropy of linear block codes from their factor graph representation. It is specially suited for sparse graphs such as those of low density parity check codes. It is based on the chain rule decomposition of the entropy and the confinement of dependencies within code subgraphs. After forcing or assuming the subgraphs to have a tree structure, the computation is done by means of density evolution as for a belief propagation analysis. We employ this method to compute upper bounds of the LDPC code syndrome entropy, which allows us to obtain asymptotic MAP upper bounds that match the ones obtained by the generalized area theorem.**

I. Introduction

Let us consider a linear code described by its parity check matrix $\mathbf{H}$ of dimensions $N-K \times N$. The rate of the code is then $R = K/N$. We denote $M = N - K$. Codewords are uniformly chosen from the codebook $\mathcal{C}$ defined by

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{F}_2^N | \mathbf{Hx} = \mathbf{0}\} \tag{1}$$

Let us assume that we transmit the codewords through a binary symmetric channel (BSC) with error probability p and capacity $C = 1-h_2(p)$, where $h_2(\cdot)$ is the binary entropy function. Let $\mathbf{y}$ be the received sequence. It can be expressed as $\mathbf{y} = \mathbf{x} \oplus \mathbf{e}$, where $\mathbf{e} \in \mathcal{F}_2^N$ is the error sequence introduced by the channel.

The mutual information $I(\mathbf{x}, \mathbf{y})$ and its complement, the equivocation $H(\mathbf{x}|\mathbf{y})$, characterize the performance of the communication system. Elias proved that linear codes can asymptotically achieve the Shannon limit, being the mutual information upper bounded by the channel capacity. However, for finite blocklength or a bounded decoding complexity, the achievable limit is reduced with respect to it. Some additional constraints on the code structure can also introduce additional losses. For example, for LDPC codes with a sparse matrix $\mathbf{H}$ with given finite degrees, Gallager already observed that the Shannon bound is not achievable [1] and proposed a simple upper bound below the channel capacity.

Burshtein et.al [2] improved Gallager's bound taking into account dependencies between pairs or clusters of syndrome bits. They also generalized the bounds to channels other than the binary symmetric channel (BSC). This was later refined in [3] defining a generalized syndrome from the received sequence, although the proposed bounds were obtained following the same approach as Gallager, neglecting dependencies among syndrome bits.

This work has been partially funded by TEC2013-47020-C2-2-R (COMPASS) (Spanish Government), 2014-SGR-60 AGAUR (Catalan Government).

In [4] Montanari proposed a method based on statistical physics to compute tight bounds on the equivocation of LDPC codes. It resembles the procedure that we propose here but it was merely employed as an interpolation mathematical tool. A more insightful procedure was proposed in [5] and other works by the same authors with the introduction of the generalized area theorem and the Maxwell decoder. Finally, coupling of low density parity check (LDPC) codes has been shown [6] to be not only a good code construction but also a procedure to prove achievable bounds on the basis of the asymptotic convergence of BP to maximum a posteriori (MAP) decoding.

This paper proposes a method to approximate the equivocation that stems from the application of the chain rule when computing the syndrome entropy, taking one step further over the approach of Gallager and Burshtein and exploiting the dependencies among syndrome bits. Unlike all cited references, it is applied over a specific parity check matrix instead of being an asymptotic approach. This can help to gain a different kind of insight in the relationship between graph structure and performance for finite blocklength that could be employed to design short codes. Nevertheless, for randomized constructions such as LDPC codes, the results coincide for any matrix from the same ensemble as long as the codeword length is not very small. This allows us to compute N-asymptotic MAP threshold bounds that match those obtained from the area theorem [5]. This brought the authors to propose an alternative asymptotic analysis based on the proposed method [9].

Let us introduce some definitions to simplify the notation. For any integer i, let $[i]$ be the set of all integers from 0 to $i-1$, i.e. $[i] = \{0, ..., i-1\}$. Let $V = \{v_0, ..., v_{|V|-1}\}$ be a subset of distinct integers from $[i]$ and $\mathbf{s}$ be a vector of length at least i. We define the mapping operation $\mathbf{s}(V)$ as: $\mathbf{s}(V) = (s_{v_0}, ..., s_{v_{|V|-1}})^T$, which corresponds to the vector with the entries of $\mathbf{s}$ indexed by the elements of V.

II. Bounds on the syndrome entropy

Let us define the syndrome $\mathbf{s}$ as the projection of the received sequence by the parity check matrix. Since $\mathbf{Hx} = \mathbf{0}$ by construction, it is equal to $\mathbf{s} = \mathbf{He}$. While this paper focuses on the BSC channel, its extension to other binary-input memoryless symmetric channels is straightforward defining a generalized syndrome as shown in [3].

The relevance of the syndrome in the system mutual information and code performance is evidenced by the following equality for the equivocation [2]:

$$H(\mathbf{x}|\mathbf{y}) = N(1-C) - H(\mathbf{s}) \tag{2}$$

which justifies the effort of analyzing the syndrome entropy.

In general, syndrome bits are not statistically independent because codeword bits participate in more than one parity check equation. Establishing an order among them, their joint entropy can be expressed following the chain rule as:

$$H(\mathbf{s}) = \sum_{i=0}^{M-1} H(s_i|\mathbf{s}([i])) \tag{3}$$

The computational complexity of evaluating the conditional statistics of s_i grows exponentially, since we are conditioning it to the value of all preceding syndrome bits $\mathbf{s}([i])$. However, depending on the code structure, the value of some syndrome bits may not be significant for the computation of the i-th contribution, or less significant than others. This motivates the introduction of the following simplification: instead of conditioning the i-th syndrome bit to all preceding ones and computing $p(s_i|\mathbf{s}([i])$, we crop the dependency to some of them, indexed by the vector $V_i^* \subset [i]$, obtaining $p(s_i|\mathbf{s}(V_i^*))$. In terms of entropy,

$$H^*(s_i) \triangleq H(s_i|\mathbf{s}(V_i^*)) \geq H(s_i|\mathbf{s}([i])) \tag{4}$$

where the inequality holds because conditioning can only reduce entropy. Adding all contributions, we obtain an upper bound of the syndrome entropy (eq. (3)):

$$H^*(\mathbf{s}) \triangleq \sum_{i=0}^{M-1} H^*(s_i) \geq H(\mathbf{s}) \tag{5}$$

The tightness of this bound depends on the selection of the sets $\{V_i^*\}$, which we call dependency sets.

III. D-Depth subgraphs

Let $\mathcal{G}$ be the bipartite graph defined by the parity check matrix $\mathbf{H}$. The left bit nodes are associated to the error sequence $\mathbf{e}$ and the right check nodes to the parity check equations and syndrome bits $\mathbf{s}$ according to the equation $\mathbf{s} = \mathbf{H}\mathbf{e}$. In order to avoid more variable definitions, we will refer to them without distinction as $\{e_j\}$ and $\{s_i\}$ for $j \in [N]$ and $i \in [M]$ respectively. The edges in the graph are indicated by the location of the ones in the matrix. Given a dependency set V_i^*, we can construct a subgraph $\mathcal{G}_i^*$ as the graph rooted in check node s_i containing the check nodes $\mathbf{s}(V_i^*)$ and all bit nodes connected to them.

The probability of s_i conditioned to all other check nodes in this subgraph is the same metric that would be required by a MAP decoder for all combinations of values of the syndrome bits. If we bound the cardinality of the dependency set, the subgraph has few check nodes and it can be computed exactly with limited complexity. A smart selection of the order and dependency sets may also help on the computational complexity and the tightness of the approximation. In particular, it is a known feature of LDPC codes and other codes with sparse parity check matrices that their factor graph is locally a tree. Hence, if we choose the dependency sets to be local around the analyzed check node, the subgraphs over which we are computing the probabilities will be trees.

Irrespective of the number of nodes, if the subgraph is a tree we know that MAP probabilities can be computed with linear complexity by means of belief propagation (BP). Moreover, in order to describe the conditioned probability of the check node s_i, we do not need to keep track of all combinations of values of the preceding check nodes but, instead, we can work over the probability distribution as done in density evolution.

Let us define the distance $d(s_i, s_j)$ between two check nodes s_i and s_j as the minimum number of bit nodes that we cross in any path between them. For example, two check nodes connected to the same bit are at distance 1. We define the dependency set of depth D of the i-th check node, V_i^D, as the set of indexes of the check nodes that are at a distance lower or equal to D from it and are precedent to it[1], i.e.

$$V_i^D = \{n \in [i] \quad | \quad d(s_i, s_n) \leq D\} \tag{6}$$

We call the subgraph spanned by those checks with s_i as root the D-depth subgraph of s_i. Note that it is dependent on the syndrome bits order.

It can be shown that if the girth of $\mathcal{G}$ is greater than $4D + 2$ then all D-depth subgraphs are trees. From now on we call them D-depth subtrees whenever we assume that this condition holds. Even if it does not, the probability of a D-depth subgraph being a tree can still be very high for many of the check nodes when D is small and the graph is sparse, so assuming that all are subtrees will lead to good approximations.

IV. DE computation of the conditional entropy

Given the D-depth subtree for the check node s_i, we can compute the probability $p(s_i|\mathbf{s}(V_i^D))$ for a given value of the check nodes in the dependency set V_i^D starting from the leaf bit nodes and propagating the probabilities to upper layers until we reach the root check node. This is the same operation performed by a BP decoder, with the only difference that the root node is a check node instead of a bit. Also, BP operates usually with log-likelihood ratios (LLRs) l instead of probabilities, which are related as

$$l = \log\left(\frac{p(s=0)}{p(s=1)}\right) \tag{7}$$

In order to fully characterize the statistics, we should do this for all values of the syndrome bits at the check nodes of the subtree. Instead, we could approximate this average by Monte-Carlo as in any code performance simulation.

Since DE computes the probability density function (p.d.f.) of the LLR messages from the BP decoder [8], we can also employ it to compute the conditional p.d.f. of the syndrome bits at the root check node of each subtree. The number of iterations performed is indicated by the subtree depth. However, instead of repeating the same bit and check node update equations at every iteration, the update is done separately for every branch according to its specific topology. This means that, in general, we need to compute and save a different p.d.f.

[1]Notice that, as for the definition of distance, we are having into account only check nodes when we talk about depth.

of the LLRs for each edge going into a node from lower to upper layers.

Since the channel error is identically distributed in all bits, only the subtree topology is significant and not the specific bit or check nodes that appear in it. This allows us to drop the identification of the nodes within the code and consider the analysis of a tree graph for which we define the following node labeling (see Figure 1):

- The tree consists of $D+1$ bit-check node layers. The j-th layer ($j \in [D+1]$) contains M_j check nodes and N_j bit nodes, being the checks the parent nodes of the bits. We enumerate the check and bit nodes by the pairs of indexes $(j,m)_c$ and $(j,n)_b$, for $m \in [M_j]$ and $n \in [N_j]$ respectively. The first layer corresponds to the root check node $(0,0)_c$ ($M_0 = 1$), while the rest of layers can have different number of check nodes or even be empty.
- Every check node $(j,m)_c$ has a parent bit from the $j-1$-th layer, indexed by $c(j,m) \in [N_{j-1}]$.
- Likewise, every bit node $(j,n)_b$ has a parent check from the same j-th layer, indexed by $b(j,n) \in [M_j]$. For example, in figure 1, $b(j, N_j - 1) = M_j - 1$. Also, since the first layer has only one check node, $b(0,n) = 0 \quad \forall n$.

Following to this notation, DE is computed recursively starting from the bit nodes of layer $j = D$ and ending at the root $(0,0)_c$. The update equations can be written as:

$$\mathrm{p}_{(j,m)_c}(l) = \circledast_{\substack{\forall n \in [N_j] \\ | \; b(j,n) = m}} \mathrm{p}_{(j,n)_b}(l) \tag{8}$$

$$\mathrm{p}_{(j,n)_b}(l) = \mathrm{p}_{BSC}(l) \circledcirc_{\substack{\forall m \in [M_{j+1}] \\ | \; c(j+1,k) = m}} \mathrm{p}_{(j+1,m)_c}(l) \tag{9}$$

for the check and bit nodes respectively. The convolution operations $\circledast$ and $\circledcirc$ are defined in [8, Ch.4] and the channel p.d.f $\mathrm{p}_{BSC}(l)$ is:

$$\mathrm{p}_{BSC}(l) = (1-p)\delta(l + L_p) + p\delta(l - L_p)$$

where $L_p = \log\left((1-p)/p\right)$ is the channel a priori LLR.

The bit nodes in the lowest layer are always leaves, so

$$\mathrm{p}_{(D,n)_b}(l) = \mathrm{p}_{BSC}(l) \qquad \forall n \in [N_D]$$

This holds also for any bit node in intermediate layers that is connected only to its parent check node.

After this procedure, the obtained p.d.f. $\mathrm{p}_{(0,0)_c}(l)$ describes the statistics of the i-th syndrome bit (root check node) conditioned to the syndrome bits from its dependency set in the original graph (rest of check nodes in the subtree). Let us rename it as $p_i(l)$ for each one of them ($i \in [M]$). The corresponding conditional entropy $H(s_i|\mathbf{s}(V_i^D))$ can be computed as

$$H(s_i|\mathbf{s}(V_i^D)) = \int_{-\infty}^{\infty} \mathrm{p}_i(l) h_2\left(\frac{e^l}{1+e^l}\right) dl \tag{10}$$

Adding them all up allows us to compute the total entropy bound as indicated in equation (5) with $H^*(s_i) = H(s_i|\mathbf{s}(V_i^D))$. Note that the computational complexity is linear with the blocklength and exponential with the depth (the base of the exponential depends on the node degrees).

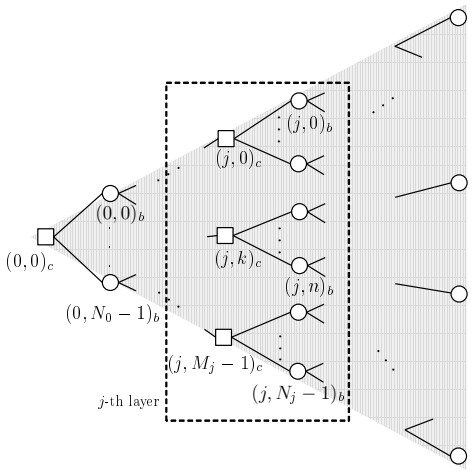

Figure 1. Subtree nodes indexation. The first layer and a generic j-th layer are represented.

V. Short code example and loopy computation

Let us consider the short code with the Tanner graph represented in figure 2a as an illustrative example. As it can be seen, it contains 9 bit and 5 check nodes, so the code rate is 0.44. The code has girth 8, as it contains a cycle comprising 4 check nodes. The check node order has been chosen arbitrarily for illustrative purposes.

We want to compute bounds on

$$H(\mathbf{s}) = \sum_{i=0}^{4} H(s_i|\mathbf{s}([i]) \tag{11}$$

For $D = 0$, we obtain:

$$H^{D=0}(\mathbf{s}) = \sum_i H(s_i) \tag{12}$$

This corresponds to minimal subtrees for all check nodes (i.e. themselves plus the connected bits), as if they were independent. There is a closed form expression for this simple case, the Gallager bound [1]. The corresponding entropies depend only on the degree of the check node, so they have the same value for all them except for s_4 (with degree 2). Figure 3 plots the evolution of the sum normalized per coded bit (dividing by $N = 9$) as a function of the channel parameter p. It is compared with $1 - C$, since their difference determines the equivocation bound (see eq. (2)). This bound is not tight: for p below 0.07 it already becomes negative even though we know that for this short blocklength the equivocation can only become zero for $p = 0$.

The case $D = 1$ is more interesting. Now, check nodes s_0 and s_1 have the same subtree as for $D = 0$, so they both have entropy $H(s_i)$ as before. Actually, this would happen not only for $D = 1$ but for any other D. It is obvious for the first check node and is also true for s_1 because it is not a direct neighbor of s_0 and, therefore, they are jointly independent. The magnitude $H(s_i)$ corresponds to their exact contribution to the syndrome entropy.

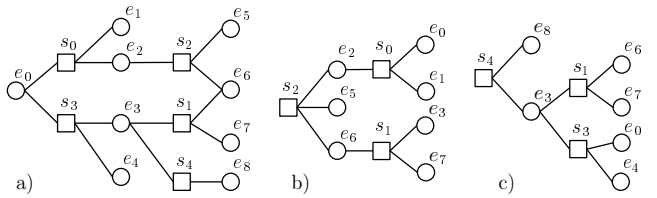

Figure 2. Example code of rate 0.44 and blocklength of 9 bits (a) and its $D = 1$-depth subtree decomposition for check nodes s_2 (b) and s_4 (c).

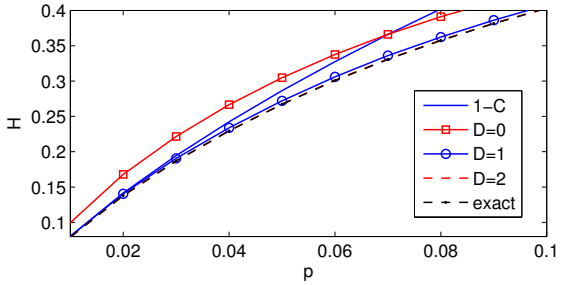

Figure 3. Syndrome entropy ('exact') and syndrome entropy bounds of the example code for depths $D = 0, 1, 2$ compared to $1 - C$ for a BSC with error probability p.

The 1-depth subtree of check node s_2 is represented in figure 2b. The subtree for s_3 has the same topology, so $H(s_2|s_0, s_1) = H(s_3|s_0, s_1)$. While the former is still the exact contribution to the entropy, the latter is already an upper bound, since we are dropping the dependence on s_2. The last check node, s_4, has the 1-depth subtree represented in figure 2c and the corresponding entropy is $H(s_4|s_1, s_3)$. We employ the described DE procedure to compute each contribution and add them up for different p values, resulting in the curve seen in figure 3. It is always smaller than $H^{D=0}$ and, for most of the plotted range of probability values, below $1 - C$. This is more in accordance with the performance we expect for this code, as it crosses it around $p = 0.014$.

For $D = 2$, we can see that the length-8 cycle appears in the subgraphs of check nodes s_3 and s_4. Actually, $D = 2$ includes all preceding check nodes, so it would lead to the computation of their exact entropy. The problem is that the subgraph is not a tree anymore and DE cannot be employed.

The exact computation of the syndrome entropy for a short code like this one can be done by direct computation of the probability of all 2^{N-K} possible syndrome values, so we also plot the result in figure 3 in order to compare it with the bounds (labeled as "exact") . Notice that the difference with the bound for $D = 1$ is small.

For larger subgraphs, the exact computation may be not feasible. However, we can deal with cycles as in a loopy belief propagation decoder: ignoring them. This is equivalent to unfolding the graph introducing new nodes so that the cycles are broken and we observe a tree, i.e. using a computation tree [7]. Figure 3 shows also the $D = 2$ entropy bound evaluated using the computation tree. Even if the difference cannot be appreciated in the used scale, It is just slightly above the exact entropy.

In general, the entropy obtained employing the computation tree when there are cycles does not guarantee the upper bound inequality for a given check node (we might get a value lower the exact contribution). However, if the graph has few short cycles they will appear in few subtrees and their entropy contribution will have no significant impact on the average. Furthermore, the larger the cycle length, the closer the behavior of the computation tree to that one of the actual subgraph. For this reason we can be almost sure that the obtained magnitude is still an upper bound and it will certainly converge for long blocklengths.

In order to guarantee the direction of the inequality, we can crop branches of the computation tree removing the repeated check nodes that were already present in preceding layers or branches. This reduces the size of the dependency sets, so it guarantees the inequality of the computed conditional entropy but at the same time it results in less tight bounds.

The complexity of the computations with cycles can be reduced also by cropping if we assume that some error bits can be recovered with high reliability from the precedent syndrome bits. This turns into the corresponding bit nodes being removed from the subgraph and, with them, any branch or cycle that had a path through them. However, this procedure introduces the opposite inequality in the computation. Let $\mathcal{W} \in [N]$ be the indexes of a subset of error bits, then:

$$H(s_i|\mathbf{s}([i])) \geq H(s_i|\mathbf{s}([i]), e(\mathcal{W}))$$

Since the dependency sets must include all precedent check nodes $\mathbf{s}([i])$, the tree topology relies now on the cropping established by the removed bit nodes. In the example, for s_3, we can obtain a lower bound from a subtree as

$$H(s_3|s_0 s_1 s_2) \geq H(s_3|s_0 s_1 s_2 e_6)$$

A smart selection of the check node order and the bit nodes to crop may allow to compute tight lower bounds on the syndrome entropy, but this remains as an open question.

VI. Bounds for LDPC codes over a BSC

Consider a regular (l, r) LDPC code, i.e. a code graph with degrees l and r for the bit nodes and check nodes respectively. The extension to irregular LDPC codes or other LDPC code ensembles is straightforward.

For any given subtree of depth D, all check nodes have degree r. The degree of the bit nodes, however, may not be regular, as it depends on how many of adjacent check nodes are in the dependency set. If none of them are, the bit node becomes a leaf. They are also leaves when their parent check node is at distance D from the root, no matter what other nodes are they adjacent to.

The subtree topology changes as we go through the ordered sequence of check nodes, going from a minimal subtree (for $i = 0$), which consists of the root node and r bit leaf nodes, to the maximal D-depth subtree (for $i = M - 1$), which happens when all the intermediate bit nodes have full degree (they are connected to $\mathrm{l} - 1$ check nodes in lower layers).

As we have seen before, the conditional probability and the entropy of the root check node depends exclusively on the topology of the subtree and not on the specific order or label of the connected bit and check nodes. Then, the average entropy bound (5) could be expressed grouping terms with equal entropy contribution as a weighted average in which the weights are the frequencies of apparition of every topology. We call these frequencies the D-depth subtree spectrum.

When the blocklength grows, it can be seen that these frequencies converge to fixed probabilities and all code realizations converge to an average ensemble spectrum. The exact description of this spectrum, however, is out of the scope of this work. This is justified because when the proposed method to compute a bound on the equivocation is applied to a specific matrix $\mathbf{H}$, we already analyze all present subtrees so we do not need an statistical description. Moreover, computations presented in this section show that, at least for low values of D, the corresponding subtree spectrum is already accurately represented in a single code realization.

We computed syndrome entropy bounds for several regular LDPC codes operating over a BSC and with $D = \{0$ (Gallager)...$3\}$. We assumed that there are no cycles and employed DE as described in section IV over the code subtrees.

We tried different parity check matrix realizations from the (l, r) regular ensemble, for $N = 1000$ and 10000 with a fixed node order, obtaining always same results. This can be explained by two reasons. On the one hand, the subtree spectrum is already well represented for short blocklengths. Even if the number of possible subtree types increases exponentially with D, the frequency of appearance of each of them decreases. Therefore, even if many types do not occur in a code realization, this happens according to their corresponding low probability.

On the other hand, subtrees with slight differences on the topology result in close entropy values, so even if the specific realization produces different subtrees, the average entropy considering similar subtrees stays practically equal. This is more noticeable for larger subtrees since their topology becomes more similar (many of the different possible subtrees are equal in the first layers).

This convergence phenomenon allows us to conjecture bounds on the MAP threshold for the code ensembles from a single parity check matrix realization of moderate blocklength applying equation (13) with no need to approach the limit.

The MAP threshold is defined as the minimum channel error probability for which the MAP decoder error probability is asymptotically bounded away from 0. Thanks to the Fano inequality it can be expressed as a function of the channel equivocation:

$$p_{MAP} \leq \min_{p} \quad \{p \quad | \quad lim_{N\to\infty} \frac{1}{N} H(\mathbf{x}|\mathbf{y}) > 0\} \qquad (13)$$

and, therefore, bounded by the syndrome entropy according to equation (2).

Results are shown in table I, where they are compared with the Shannon bound ($C^{-1}(R)$) and with the bound proposed in [4], which happens to be very close to the 1-depth bound. They numerically coincide with the ones computed from the area theorem [6, tab.II]. As expected, the bound decreases when D grows, as more checks are included in the dependency sets. Note that the reduction ameliorates when D grows and, eventually, it will converge to a fixed value (the tightest bound). In most cases this convergence occurs for low values of D (like $D = 1$ for the $(4, 6)$ code). This happens because, when the channel error probability is above the BP threshold, the BP decoder converges to a fixed point corresponding to a non-zero entropy at the root check nodes. The further we are above the BP threshold, the fastest - in number of iterations or subtree depth - the fixed point is reached. This is bad under BP decoding point of view, as coded bits cannot be recovered, but it is good under the point of view of syndrome entropy since it guarantees that all syndrome bits provide information.

l, r	$D = 0$	$D = 1$	$D = 2$	$D = 3$	[4]	Sh
$3, 6$	0.1025	0.1007	0.1001	0.099	0.101	0.11
$3, 5$	0.1397	0.1382	0.1376	0.1374	0.1384	0.146
$4, 6$	0.1726	0.1725	0.1724	0.1724	0.1726	0.174
$4, 8$	0.1076	0.1073	0.1072	0.1072	-	0.11

Table I
UPPER BOUNDS ON THE CHANNEL ERROR PROBABILITY OF A BSC FOR RELIABLE TRANSMISSION WITH (l, r) REGULAR LDPC CODES.

VII. CONCLUSIONS

In this paper, we propose a method for approximating the syndrome entropy of linear block codes based on the chain rule decomposition and the simplification of the computation by cropping dependencies. For sparse parity check matrices, like those of LDPC codes, the computation can be performed by means of DE over subtrees of different topologies. We describe the procedure and then apply it in order to obtain asymptotic bounds on the MAP threshold for regular LDPC codes thanks to their fast convergence with the blocklength.

REFERENCES

[1] R.G. Gallager, "Low- Density Parity- Check Codes," *M.I.T. Press*, Cambridge, Massachusetts, 1963.

[2] D. Burshtein, M. Krivelevich, S. L. Litsyn, and G. Miller, "Upper bounds on the rate of LDPC codes," *IEEE Trans. Inform. Theory*, vol. 48, no. 9, pp. 2437-2449, Sept. 2002.

[3] G. Wiechman and I. Sason, "Parity-check density versus performance of binary linear block codes over memoryless symmetric channels: New bounds and applications," *IEEE Trans. Inform. Theory*, vol. 53, no. 2, pp. 550-579, February 2007.

[4] A. Montanari, "Tight bounds for LDPC and LDGM codes under MAP decoding," *IEEE Trans. Inform. Theory*, vol. 51, no. 9, pp. 3221-3246, Sept. 2005

[5] C. Méasson, A. Montanari, and R. Urbanke, "Maxwell's construction: The hidden bridge between maximum-likelihood and iterative decoding," *Int. Symp. on Inform. Theory*, 2004.

[6] S. Kudekar, T. Richardson, R. Urbanke, "Spatially coupled ensembles universally achieve capacity under belief propagation," *Int. Symp. on Inform. Theory* , 2012.

[7] N. Wiberg, "Codes and Decoding on General Graphs," *Ph.D. thesis*, Linköping University, Sweden, 1996.

[8] T. Richardson and R. Urbanke, "Modern Coding Theory," *Cambridge University Press*, 2008.

[9] D. Matas and M. Lamarca, "Asymptotic MAP upper bounds for linear block codes," *Int. Symp. on Inform. Theory* , 2016.