HAL open science

lambda-Min Decoding Algorithm of Regular and Irregular LDPC Codes

Emmanuel Boutillon, Frédéric Guillou, Jean-Luc Danger

To cite this version:

Emmanuel Boutillon, Frédéric Guillou, Jean-Luc Danger. lambda-Min Decoding Algorithm of Regular and Irregular LDPC Codes. 3nd International Symposium on Turbo Codes & Related Topics, 2003, Brest, France. hal-00068941

HAL Id: hal-00068941

https://hal.science/hal-00068941v1

Submitted on 15 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

$\lambda-$Min Decoding Algorithm of Regular and Irregular LDPC Codes

Frédéric Guilloud [(1)], Emmanuel Boutillon [(2)], Jean-Luc Danger [(1)]

(1) Telecom Paris, 46 rue Barrault, 75634 PARIS Cedex 13, FRANCE
E-mail: frederic.guilloud@enst.fr, jean-luc.danger@enst.fr
(2) LESTER, Université de Bretagne Sud, BP 92116, 56321 LORIENT Cedex FRANCE
E-mail: emmanuel.boutillon@univ-ubs.fr

Abstract: *In this paper, the decoding of low-density parity-check (LDPC) codes is considered. A new algorithm, named $\lambda-$Min algorithm, for updating extrinsic information is proposed. The $\lambda-$Min algorithm offers different trade-off performance versus complexity between the belief propagation (BP) algorithm (optimal but complex) and the universal most powerful (UMP) BP-based algorithm (simple but leading to significant performance degradation in some cases). Hardware implementation of the $\lambda-$Min algorithm in a serial mode is also discussed. Reduction up to 75 % of the extrinsic memory information can be obtained with high rate LDPC code without any significant degradation of the performance.*

Keywords: Iterative decoder, LDPC Codes, UMP BP-Based Algorithm, $\lambda-$min algorithm

1. INTRODUCTION

With the choice of LDPC (low-density parity-check) codes as a standard for DVB-S2, VLSI architectures for LDPC code decoders become a real challenge. In order to decrease the complexity of the decoding algorithm, Fossorier and al. proposed simplified versions of the Belief Propagation (BP) algorithm named BP-Based, offset BP-based [1], [2], [3]. These algorithms are efficient for regular LDPC codes with small length, but they introduce some significant degradation (up to almost 1 dB) for LDPC codes with high degree and high length. In this paper, we propose a new decoding algorithm named $\lambda-$min algorithm which offers a complexity-performance trade-off between BP (optimal) and BP-based algorithm. Moreover, we study the VLSI implementation of the sub-optimal algorithm and we propose a parity check processor which enables to compute efficiently the $\lambda-$min algorithm and which reduces the memory required to save extrinsic information between two decoding iterations. The rest of the paper is organized as follows. In section 2, the BP, the BP-based and the new $\lambda-$min algorithms are described. Simulation results are given in section 3 and an optimization of the $\lambda-$min algorithm is proposed. Section 4 describes an efficient serial architecture to process the $\lambda-$min algorithm and a conclusion is given in section 5.

2. ITERATIVE DECODING ALGORITHMS

In the following, a codeword $x = (x_0\, x_1\, \cdots\, x_{N-1})$ is transmitted with a BPSK modulation scheme over an AWGN channel. The received bits are then $y_n = -(-1)^{x_n} + b_n$ where $b_n \sim \mathcal{N}(0, \sigma^2)$. The signal to noise ration (SNR) will refer to the ratio E_b/N_0, where E_b is the energy per bit before coding, and $N_0/2 = \sigma^2$. The parity-check matrix H is an $M \times N$ regular or irregular sparse matrix.

According to the notations of [1] and [2], $\mathcal{N}(m) = \{v_n : H_{mn} = 1\}$ denotes the set of bits v_n that participate in parity check c_m and $\mathcal{N}(m)\backslash n$ denotes the same set with bit v_n excluded. Similarly, $\mathcal{M}(n) = \{c_m : H_{mn} = 1\}$ denotes the set of parity check c_m in which the bits v_n participate and $\mathcal{M}(n)\backslash m$ denotes the same set with parity check c_m excluded. The cardinality of a set $\mathcal{A}$ is denoted by $|\mathcal{A}|$. $Z_{mn}^{(i)}$ denotes the Log-Likelyhood Ratio (LLR) of the information which is sent from v_n to c_m in the ith iteration. $L_{mn}^{(i)}$ denotes the LLR of the information which is sent from c_m to v_n in the ith iteration.

2.1. Belief propagation decoding:

BP decoding [4] is the optimal iterative algorithm to achieve capacity-approaching performance with LDPC codes. It is based on the propagation of the bit information v_n, $n \in \{0, \cdots, N-1\}$ through the parity-checks c_m, $m \in \{1, \cdots, M-1\}$:

Initialization: A-priori information is initialized by the LLR of y_n: $L_n^{(0)} = \frac{2y_n}{\sigma^2}$ and set all the $L_{mn}^{(0)}$ to zero.

Iterations: An iteration i has 3 steps:

1. Update bits: for each v_n and each $c_m \in \mathcal{M}(n)$, compute:

$$Z_{mn}^{(i)} = Z_n^{(i)} - L_{mn}^{(i-1)} \tag{1}$$

with $Z_n^{(i)} = L_n^{(0)} + \sum_{m\in\mathcal{M}(n)} L_{mn}^{(i)}$ being the soft values of bits v_n. Calculate also the syndrome $s_i(\hat{x}) = H\hat{x}_i'$ of the estimated received codeword $\hat{x}_i = \left\{ sign\left(Z_n^{(i)}\right)\right\}_{1\le n\le N}$.

2. Update parity-checks: for each c_m and each $v_n \in \mathcal{N}(m)$ compute:

$$L_{mn}^{(i)} = S_{mn}^{(i)} \times M_{mn}^{(i)} \tag{2}$$

with $S_{mn}^{(i)} = \text{sign}\left(Z_{mn}^{(i)}\right) \prod_{n \in \mathcal{N}(m)} \text{sign}\left(-Z_{mn}^{(i)}\right)$

and

$$M_{mn}^{(i)} = -\bigoplus_{n \in \mathcal{N}(m)\backslash n} \left(-\left|Z_{mn}^{(i)}\right|\right) \tag{3}$$

and $\bigoplus_n (I_n) = I_0 \oplus I_1 \oplus \cdots \oplus I_n$, where $\oplus$ is the commutative and associative function such that [5]:

$$I_0 \oplus I_1 = \ln \frac{\exp(I_1) + \exp(I_2)}{1 + \exp(I_1 + I_2)} \tag{4}$$

3. Stop the iterations if $i = i_{\max}$ or if $s_i(\hat{x}) = 0$.

Iterative BP-based algorithm for LDPC codes decoding has been proposed by Fossorier and al. [1]. It simplifies the check-node processing, and does not need any knowledge of the channel: initialization is replaced by $L_n^{(0)} = y_n$ and (3) is replaced by:

$$M_{mn}^{(i)} = \min_{n \in \mathcal{N}(m)\backslash n} \left|Z_{mn}^{(i)}\right| \tag{5}$$

The degradation of the Bit Error Rate (BER) introduced by the BP-based algorithm can be very significant. In fact, equation (5) over estimates the LLR of the extrinsic information. This over-estimation can be partially corrected by the addition of an offset factor (offset BP-based algorithm [2], [3]) but still, for near shannon limit LDPC codes (irregular LDPC codes with high length), the offset BP-based algorithm introduces significant performance degradation (up to 1 dB of SNR).

2.2. $\lambda-$Min Algorithm

We present here a more accurate simplification, based on the BP algorithm. Equation (4) can be decomposed as the sum of 3 terms:

$$\begin{aligned} I_0 \oplus I_1 &= -\text{sign}(I_0)\,\text{sign}(I_1)\min(|I_0|, |I_1|) \\ &+ \ln\left(1 + \mathrm{e}^{-|I_0 - I_1|}\right) - \ln\left(1 + \mathrm{e}^{-|I_0 + I_1|}\right) \end{aligned} \tag{6}$$

If $I_0 >> I_1$, the two last correction factor of (6) are negligible, and thus $I_0 + I_1 \simeq I_0$. More generally, the result of (3) depends mainly on the smallest values of $Z_{mn}^{(i)}$. In order to simplify the computation of this equation, we propose to compute (3) with only the λ lowest magnitude of extrinsic information, with $\lambda > 1$. Let $\mathcal{N}_\lambda(m) = \{n_0, \cdots, n_{\lambda-1}\}$ be the subset of $\mathcal{N}(m)$ which contains the λ bits of parity check c_m which LLR have the smallest magnitude. Equation (3) is then approximated by:

$$M_{mn}^{(i)} = -\bigoplus_{n' \in \mathcal{N}_\lambda(m)\backslash n} -\left|Z_{mn'}^{(i)}\right| \tag{7}$$

Two cases will occur: if the bit v_n belongs to the subset $\mathcal{N}_\lambda(m)$, then the $Z_{mn}^{(i)}$ are processed over $\lambda - 1$ values of $\mathcal{N}_\lambda(m)\backslash n$, otherwise the $Z_{mn}^{(i)}$ are processed over the λ values of $\mathcal{N}_\lambda(m)$. Hence, for the second case, the computation have to be performed only once.

Note that the case $\lambda = 2$ differs only from the BP-based algorithm by the approximation of $Z_{mn}^{(i)}$ in the case of $n \notin \mathcal{N}_2(m) = \{n_0, n_1\}$: the BP-based uses $Z_{mn_0}^{(i)}$ whereas $2-$min uses $Z_{mn_0}^{(i)} \oplus Z_{mn_1}^{(i)}$.

3. SIMULATION RESULTS

Simulations have been performed using two different codes: $\mathcal{C}_1$ is a regular (5,10) LDPC code of length $N = 816$, from [6] and $\mathcal{C}_2$ is an irregular LDPC code of length $N = 2000$ and rate $R = 0.85$. Its distribution degree is taken from [7], code number 325.

3.1. Algorithm Comparison

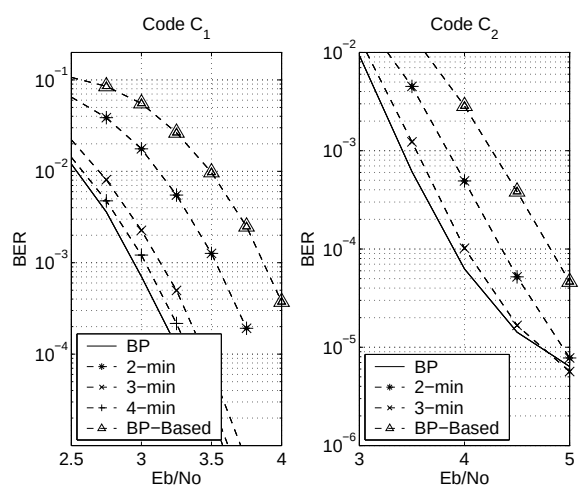

Figure 1: Performance of $\lambda-$Min algorithm with $\lambda = \{2, 3, 4\}$ for code $\mathcal{C}_1$ and code $\mathcal{C}_2$ (50 iterations max).

Figure 1 compares the performance of $\mathcal{C}_1$ and $\mathcal{C}_2$ using respectively the BP, the $\lambda-$min and the BP-based algorithms, all with 50 decoding iterations. As expected, the $\lambda-$min algorithm outperforms the BP-based algorithm and gets closer to the BP algorithm as λ increases. Using the $3-$min algorithm for the code $\mathcal{C}_2$, which have check node degree of 40 or 41, introduces a small degradation of 0.15 dB at a BER of 10^{-4} (instead of 0.9 dB for BP-based) and can save up to 67% of memory compared to the BP algorithm (see figure 5 in section 4).

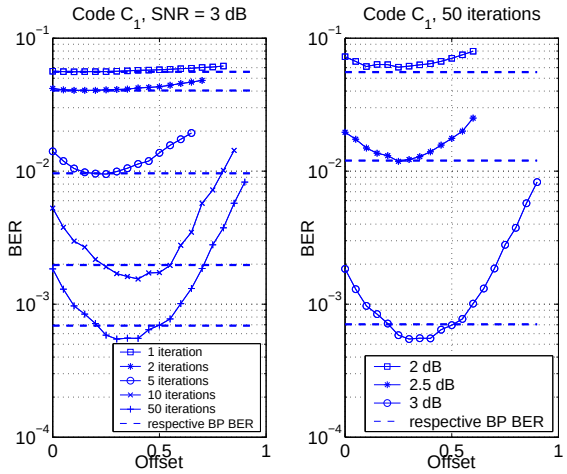

Figure 2: Influence of the offset value on the BER for the 3−min algorithm, for several SNR and several max iteration number, for $\mathcal{C}_1$. The BER obtained with the BP algorithm is represented with a dashed line. Similar results stand for code $\mathcal{C}_2$.

3.2. Optimization

As shown in figure 1, the $\lambda-$min algorithm improves the performance compared to the BP-based algorithm but there is still a degradation compared to the BP algorithm. This degradation can be reduced by the addition of an offset, as proposed by [3] in the case of BP-Based algorithm: the offset compensates the over estimation of extrinsic information given by (7). Equation (2) is then replaced by:

$$L_{mn}^{(i+1)} = S_{mn}^{(i)} \times \max\left(M_{mn}^{(i+1)} - \beta, 0\right), \ \beta > 0. \quad (8)$$

In a practical case (finite code length and finite number of iterations) the optimal value of β can simply be found by simulations.

Figures 2 depicts the evolution shape of the BER as a function of β, for several SNRs and for several max iteration number. The maximum number of word errors is only 50, within a maximum of 10^5 transmitted codewords. One can note that the offset 3−min algorithm outperforms the BP algorithm. This is explained by a faster convergence; when the number of iterations increases (figure 3), the two algorithm perform almost identically.

4. ARCHITECTURAL ISSUES

The detailed architecture of the $\lambda-$min algorithm is not described in this paper. We just give an example of the scheduling that enables to perform the $\lambda-$min algorithm very efficiently and to decrease the amount of memory needed to store the extrinsic information.

The architecture is based on a serial-parallel-serial scheduling. Each decoding iteration requires the com-

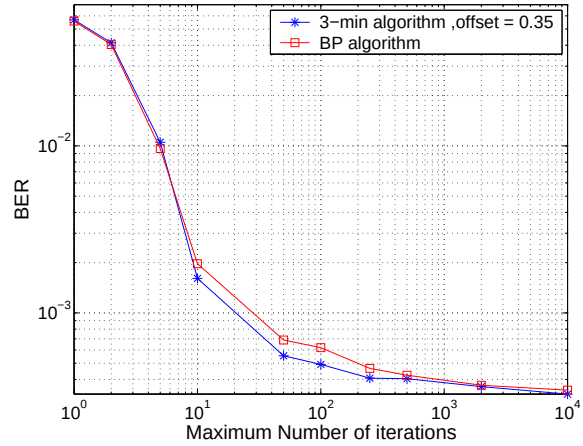

Figure 3: Comparison between BP algorithm and offset 3−min algorithm as a function of the number of iterations for a fixed SNR of 3 dB.

putation of M parity check equations. A decoding iteration is divided into M/P macro cycles. In a macro cycle, P Parity Checks Processors (PCP) perform serially the computation of P parity checks. Note that this architecture is a modification of [8]. A detailed architecture is given in [9].

Each PCP of iteration i works in 3 steps:

1. *Generating the $Z_{mn}^{(i)}$ values:* during the first $|\mathcal{N}(m)|$ clock cycles, the PCP associated to the parity check c_m received serially the $Z_n^{(i)}$, $n \in \mathcal{N}(m)$, from an external memory. At cycle n, the $L_{mn}^{(i-1)}$ of the previous decoding iteration is retrieved from an internal memory of the PCP and subtracted to $Z_n^{(i)}$ in order to generate $Z_{mn}^{(i)}$ according to (1).

2. *Sorting of the $Z_{mn}^{(i)}$:* the sorting of the $Z_{mn}^{(i)}$ value is performed serially by an insertion technique. Every clock cycle, the current value of $Z_{mn}^{(i)}$ is compared to the λ previous value, which are stored with their index in a register file of size λ. According to the result of the comparators, the new value is inserted in the sorting order in the register and the highest values is forgiven. Meanwhile, the product of all signs is also computed.

3. *Generating extrinsic information:* the extrinsic values are computed on the fly, one at each clock cycle, from the $\mathcal{N}_\lambda(n)$ values stored in the register file. Figure 4 gives an example of a possible realization for $\lambda = 3$. When the current cycle correspond to a value of $\mathcal{N}_\lambda(n)$, the corresponding boolean is set to $+\infty$ in order to bypass the XOR function that perform (7) in order to compute (8).

In order to save memory, the information needed to

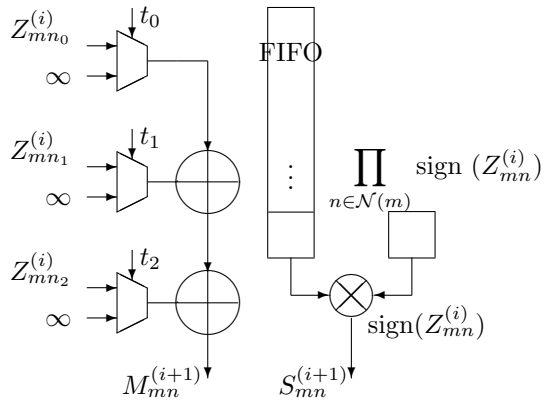

Figure 4: On the fly parity check scheme for $\lambda = 3$. When the reading phase is over, λ values are saved and are addressed via $t_0, \cdots, t_{\lambda-1}$ by the writing phase for on the fly LLR computing.

Table 1: Details of the memory required for saving extrinsic information for each parity check with the $\lambda-$min algorithm. $\lceil . \rceil$ denotes the ceil function.

Information to be stored	Number of bits
$\lambda + 1$ results of (7)	$(\lambda + 1)N_b$
λ index of $\mathcal{N}_\lambda(m)$	$\lambda\lceil \log_2(\lvert\mathcal{N}(m)\rvert)\rceil$
$\lvert\mathcal{N}(m)\rvert$ signs + product of signs	$\lvert\mathcal{N}(m)\rvert + 1$
Storage of all $Z^{(i)}_{mn}$	$(N_b + 1)\lvert\mathcal{N}(m)\rvert$

recompute the $Z^{(i)}_{mn}$ are stored in the PCP, instead of saving the $|\mathcal{N}(m)|$ values of $Z^{(i)}_{mn}$. Table 1 gives the amount of memory in the 2 cases, assuming that the magnitudes of $Z^{(i)}_{mn}$ are coded on N_b bits. Figure 5 shows the memory reduction factor obtained for $N_b = 5$. For $|\mathcal{N}(m)| = 40$ and $\lambda = 3$, the proposed solution requires only 33% of memory compared to the classical solution, which means 67% of memory saved.

5. CONCLUSION

In this paper, the decoding of LDPC codes has been considered. A new algorithm, named $\lambda-$Min algorithm, for updating extrinsic information has been proposed. The $\lambda-$Min algorithm offers different trade-off performance versus complexity between the BP algorithm and the BP-based algorithm. Simulation results for a $(5, 10)$ regular LDPC code of length $N = 816$ and for an irregular LDPC code of high rate $r = 0.85$ and of length $N = 2000$ show that the degradations introduced by the $\lambda-$min algorithm are below 0.15 dB for $\lambda \geq 3$. Moreover, the addition of an offset increases the convergence speed of the $\lambda-$min algorithm: for a given number of iterations, it can outperform the BP algorithm. A hardware realization of the $\lambda-$Min algorithm in a serial mode has been also discussed. Reduction up to 75 % of

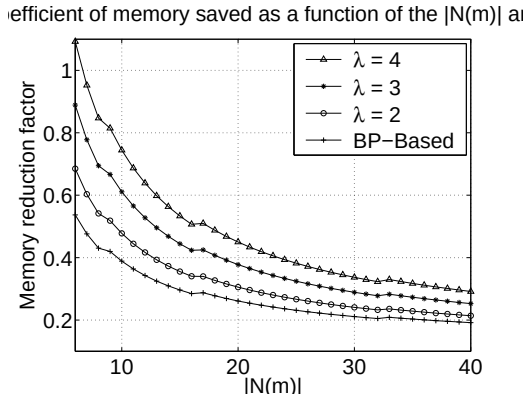

Figure 5: Evolution of the memory ratio of the 2 way of saving extrinsic information, depending on the parity check degree, for several λ value. $N_b = 5$ bits

the extrinsic memory information can be obtained with high rate LDPC code without any significant degradation of the performance.

REFERENCES

[1] M.P.C. Fossorier, M. Mihaljević, and I. Imai. Reduced complexity iterative decoding of low-density parity-check codes based on belief propagation. *Transactions on Communications*, 47:673–680, May 1999.

[2] J. Chen and M.P.C. Fossorier. Near optimum universal belief propagation based decoding of low-density parity-check codes. *Transactions on Communications*, 50:406–414, March 2002.

[3] J. Chen and M.P.C. Fossorier. Density evolution for two improved bp-based decoding algorithms of ldpc codes. *IEEE Communication Letters*, 6:208–210, May 2002.

[4] D.J.C. Mackay. Good error-correcting codes based on very sparse matrices. *Transactions on Information Theory*, 45:399–431, March 1999.

[5] G. Battail and A. H. M. El-Sherbini. Coding for radio channels. *Annales des Tlcommunications*, 37:75–96, 1982.

[6] D.J.C. Mackay. Online database of low density parity check codes. Available at http://www.inference.phy.cam.ac.uk/mackay/codes/data.html.

[7] R. Urbanke. Ldpcopt. Available at http://lthcwww.epfl.ch/research/ldpcopt/.

[8] E. Boutillon, J. Castura, and F.R. Kschichang. Decoder-first code design. In *Proceedings of the 2nd International Symposium on Turbo Codes and Related Topics*, pages 459–462, Brest, France, 2000.

[9] F. Guilloud, E. Boutillon, and J.-L. Danger. Submitted to *19e colloque GRETSI sur le traitement du signal et des images*, 2003.