An Approximate Hardware Check Node for λ-min-based LDPC Decoders

Georgios Perris-Samios and Vassilis Paliouras
Department of Electrical and Computer Engineering
University of Patras, Patras, Greece
Email: ece7362@upnet.gr, paliuras@ece.upatras.gr

***Abstract*—In this paper, iterative decoding using Belief Propagation λ-min decoding algorithm is considered. In this algorithm check nodes use only the λ lowest-magnitude messages thus simplifying the hardware complexity and reducing memory usage. A parallel-input architecture is proposed for the check node. We focus on the determination of the sought minima in a parallel fashion. Novel simplified circuits for the derivation of the λ minimum values are introduced here. The main novelty that leads to substantial hardware simplification is the approximate derivation of the λ values. Specifically, We here show that using the introduced approximate computation substantial hardware savings are obtained with no significant degradation in decoding performance. For cases of practical interest the proposed solution is shown to reduce the number of comparisons per check node from 14 down to 7; i.e., 2 times.**

***Index Terms*—Low density parity-check codes (LDPCc), Belief propagation decoding algorithm (BP), Belief propagation λ-min, λ-min approximation.**

I. Introduction

LDPC codes, initially discovered by Gallager [1] and later re-invented, have proliferated in applications and have been adopted in several international standards, such as Wi-Fi IEEE 802.11n/ac, digital television (DVB-S2/T2/C2), ITU-T G.hn/G.9960 for power line communications, CMMB, IEEE 802.3an for 10 Gbps Ethernet over twisted pair, etc. LDPC codes offer strong error correction that can be exploited by iterative decoding, which renders efficient decoders feasible. To improve the efficiency of iterative decoders, several LDPC iterative decoding algorithms have been proposed. Optimal decoding is achieved by the so-called sum-product (SP) algorithm and its implementation in the logarithmic domain, the log-SP. To further reduce complexity, several approximations to log-SP have been proposed in the literature. Prominent among them are the min-sum algorithm, the normalized min-sum, and the offset min-sum.

In this paper we propose simplified networks for the approximate derivation of the λ-minimum values out of the messages that enter a check node. We show that the approximate computation has a minor impact on the decoding performance of the λ-min algorithm [2]; however it substantially reduces the hardware complexity of the check node. The proposed approach is useful as it facilitates the implementation of decoders with check nodes that receive messages in parallel. In fact the proposed method reduces the sorting complexity by more than two times. Hence the design of highly parallel decoders based on λ-min algorithm becomes possible.

The remainder of the paper is organized as follows: Section II reviews basics of the iterative decoding algorithms. Section III introduces the proposed approximate technique, while Section IV quantifies hardware savings achieved in the check node. Finally, conclusions are discussed in Section V.

II. Iterative decoding and the λ-min algorithm

A. Belief Propagation algorithm

Belief propagation (BP) algorithm, also called sum-product message-passing, has been proposed for decoding LDPC codes. BP is an iterative algorithm with optimal performance that approaches Shannon's limit [3] and it is detailed below. Assume that the Parity Check matrix of the corresponding LDPC code is H, Channel LLRs are y_n, $n \in \{0, 1, .., N-1\}$, Variable nodes are u_n, $n \in \{0, 1, .., N-1\}$, Check nodes are c_m, $m \in \{0, 1, .., M-1\}$, Check-to-Bit messages are $\mathrm{L}_{mn}^{(i)}$, Bit-to-Check messages are $\mathrm{Z}_{mn}^{(i)}$, Soft values of variables are $\mathrm{Z}_n^{(i)}$, and the Initializing information is $\mathrm{L}_n^{(0)} = 2\frac{y_n}{\sigma^2}$ This is used in the first iteration as $\mathrm{Z}_n^{(i)}$. Exponent (i) indicates computation at the ith iteration. The Iterative process is as follows:

1) Check-to-Bit messages update: for all c_m and the u_n connected to them we compute.
$$S_{mn}^{(i)} = \mathrm{sign}(Z_{mn}^{(i)}) \times \prod_{n \epsilon N(m)} \mathrm{sign}(-Z_{nm}^{(i)}) \quad (1)$$
For all the variables connected except the one we compute, its message is derived as:
$$M_{mn}^{(i)} = -\bigoplus(-|Z_{mn'^{(i)}}|) \quad (2)$$
where symbol $\bigoplus$ denotes the commutative and associative function [4]:
$$I_1 \oplus I_2 = \frac{\exp(I_1) + \exp(I_2)}{1 + \exp(I_1 + I_2)} \quad (3)$$
with $\oplus(I_n) = I_0 \oplus I_1 \ldots \oplus I_n$. From (1) and (2), Check-to-bit messages are obtained as
$$L_{mn}^{(i)} = S_{mn}^{(i)} \times M_{mn}^{(i)} \quad (4)$$
2) Bit-to-Check message update: for all u_m and each c_m connected to them we compute $Z_n^{(i)} = L_{mn}^{(0)} + \sum_{m \epsilon M(n)} L_{mn}^{(i)}$and $Z_{mn}^{(i)} = Z_n^{(i)} - L_{mn}^{(i-1)}$
3) For stopping criteria compute: $\hat{x_i} = \mathrm{sign}(Z_n^{(i)})$ and $s_i(\hat{x_i}) = H \times \hat{x_i}$. If $s_i(\hat{x}) = 0$, the decoded codeword is correct and decoding stops with success. If it is not, operation proceeds to the next iteration. The procedure terminates when iterations reach a predefined number.

ISBN 978-0-9928626-7-1

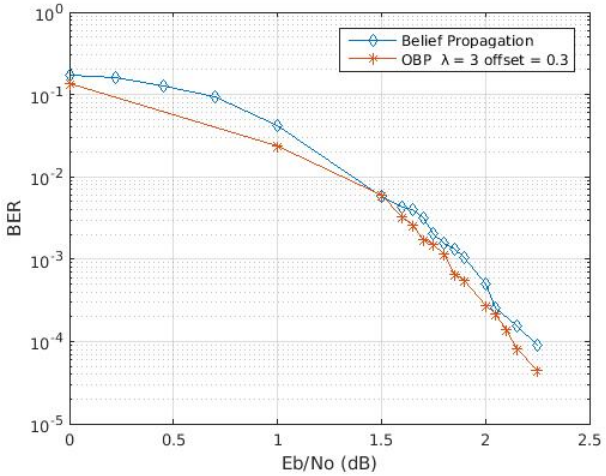

Fig. 1. Comparison of BP and BP λ min with offset

B. Simplified BP algorithms and BP λ-min decoding

Many simplifications of the complex check-node process in BP have been proposed such as [5] or the well-known BP-based algorithm proposed in [6]. This algorithm is simple but there is significant degradation in performance.

An interesting simplification is proposed in [7] called BP λ-min. Equation (3) can be written:

$$I_0 \oplus I_1 = -\operatorname{sign}(I_0)\operatorname{sign}(I_1)\min(|I_0|\,|I_1|) + \ln(1+\exp(-|I_0 - I_1|)) - \ln(1+\exp(-|I_0 + I_1|)). \quad (5)$$

In the λ-min approach, the check nodes produce messages only by using λ lower-magnitude values of messages received, with $\lambda > 1$. A new subset is created, namely $N_\lambda(m) = (n_0, n_1.., n_{\lambda-1})$, which contains these λ values. Subsequently, (2) is computed using only the limited set of approximated values. In this case if the bit processed belongs to the $\mathrm{N}_\lambda(m)$, (2) uses only the $\lambda - 1$ values of the subset, otherwise it uses all values in the subset $\mathrm{N}_\lambda(m)$. Furthermore an offset β can be used to reduce the performance loss over BP [2]:

$$L_{mn}^{(i)} = S_{mn}^{(i)} \times \max(M_{mn}^{(i)} - \beta, 0), \beta > 0 \quad (6)$$

Optimal determination of β is achieved experimentally. Assuming a code rate-1/2 648-bit LDPC code used in WiFi, with check degrees 7 and 8, Fig. 1 compares BP and λ-min decoding in terms of BER *vs.* noise level behavior after a maximum of 50 iterations, using $\lambda = 3$ and $\beta = 0.3$.

III. Proposed approximate ordering

The hardware architecture of the check node for the BP λ-min algorithm [2] is a serial-input component, as messages enter one-by-one, one per clock, as shown in Fig. 2. This architecture cannot be directly employed in a parallel decoder requiring check nodes that receive messages in parallel. Although there are drawbacks, a serial mode architecture is less complex and thus it is up to the designer to select the most appropriate implementation exploring trade-offs between timing and hardware complexity.

When a parallel-input approach is opted for the check node, there is a huge variety of sorting networks that can be used for

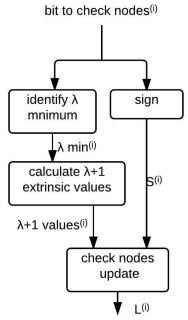

Fig. 2. Architecture of Check node process

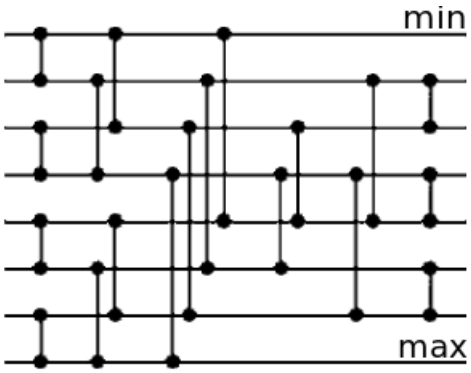

Fig. 3. Knuth diagram for sorting 8 messages

the identification of the λ minima, one of the tasks of the check node as shown in Fig. 2. The check node performs three tasks. Block sign update defines the total sign using all variables connected to the certain check node. Block λ-minimum defines the λ lowest magnitude values and is detailed below. The other block calculates, with the help of look up tables, the λ+1 extrinsic values used by the check node along with the signal to result in the final values for each check node message to the variables. The exponent (i) indicates the current iteration of the decode operation.

For the particular experiments an irregular code rate-1/2 648-bit LDPC code with check degrees of 7 and 8 is assumed. The sorting network we use [8] to identify the λ-minimum values is shown in the Knuth diagram in Fig. 3. In Knuth diagrams, each time two messages are compared, the minimum value is streamed to the upper wire and the maximum value to the lower. For this particular code, the operation demands the use of 19 comparators for the full sort and 14 comparators for the sort of the λ-minimum values of information, $\lambda = 3$. The higher the check degree is, the complexity of the network increases. Due to the complexity of this operation and motivated by an efficient approximation used for the min-sum algorithm [9], the approximation of the λ-minimum values of the check messages set is proposed.

A. Proposed approximation using 13 comparisons

The first approximation proposed is depicted in Fig. 4 and it utilizes 13 comparators. This network can always identify correctly the exact two minimum values from the set of the input messages but the third value is approximated. At first

ISBN 978-0-9928626-7-1 © EURASIP 2017

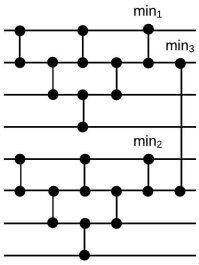

Fig. 4. Approximation using thirteen comparisons

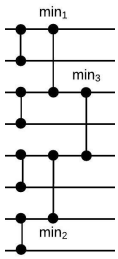

Fig. 5. Approximation using seven comparisons

we split the received messages into the upper and the lower subset of four messages. After ten comparisons are performed, we have defined the two minimum values of the each subset. The next stage uses two comparisons to identify the first and the second minimum values. The next comparison defines the approximation of the third minimum value. This operation identifies all three minimum values correctly when they do not belong in the same subset, upper or lower.

For example, assuming a set with a total of 8 messages $S_{in} = \{2,4,7,1,9,3,5,8\}$. The upper subset is $S_u = \{2,4,7,1\}$ and the lower $S_l = \{9,3,5,8\}$. After the first 10 comparisons we have $S_{u10} = \{2,1\}$ and $S_{l10} = \{3,5\}$ which are the minimum values of the subsets. The minimum messages of each pair are the first and the second minimum of the input set $S_2 = \{1,3\}$ and with the comparison of the maximum values of the subsets S_{u10} and S_{l10} we define the approximated third value. Therefore the sought three minima are $S_m = \{1,2,3\}$. In this case these are the actual minima of the input set too. As a second example, assume the set $S_{in} = \{1,5,3,2,9,8,4,7\}$. Following the same procedure the result is $S_m = \{1,4,2\}$, so we obtain 4 as third minimum instead of number 3.

Using this network in BP λ-min with offset decoder for the identification of the λ-minimum values used in the check node, the decoding performance degradation is found to be negligible compared to the same decoding algorithm using a full sorting operation. The comparison between these alternatives can be seen in Fig. 6 in terms of Bit Error Rate (BER) *vs.* noise level for a maximum of 50 decoding iterations. The optimal offset may differ for the decoder using the approximated minima and should be defined with trials.

B. Proposed approximation using seven comparisons

In Fig. 5 the Knuth diagram of the second approximation proposed that utilizes seven comparisons is shown. This network can also identify correctly the two minimum values from the input messages but has reduced probability to correctly identify the third, than the previous approximation. A comparison of the two proposed networks is shown in Table I in terms of the probability to correctly identify minima. In this network messages are initially compared in pairs. Afterwards the minimum results of each comparison are again compared in pairs thus resulting in obtaining the two minimum values of the input set in the second stage. Finally the maximum results of the second stage are compared to define the third approximated minimum value.

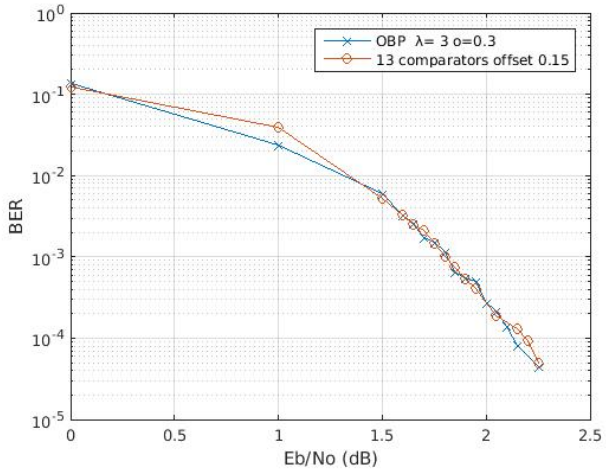

Fig. 6. Comparison of full sort with 13 comparator network (Approx1)

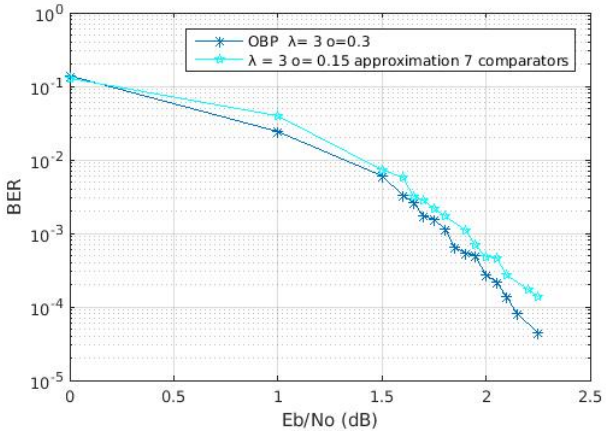

Fig. 7. Comparison of full sort with 7-comparator network (Approx2)

TABLE I
NETWORKS COMPARISON

	Approx1	Approx2
Three minima correct	86%	57%
Two minima correct	24%	43%

Although this approximation has a low probability to identify all actual three minimum messages, quantified at around 57%, if it is used with a BP λ-min decoding algorithm with offset, we here find that there is only a minor degradation of 0.1 dB in the coding gain. Fig. 7 shows this comparison with a BER *vs.* noise plot assuming the same code as before.

IV. PROPOSED ARCHITECTURES AND EVALUATION

Each one of the networks proposed, uses comparators denoted as C_{2-1} (Fig. 9) and C_{2-2} (Fig. 10). The first identifies only the minimum value among the inputs given, while the second one outputs both the minimum and the maximum value. The first network is implemented as shown in Fig. 11 and uses eight C_{2-2} and five C_{2-1} comparators. The second network requires two C_{2-2} and five C_{2-1} comparators as shown in Fig. 12.

ISBN 978-0-9928626-7-1 © EURASIP 2017

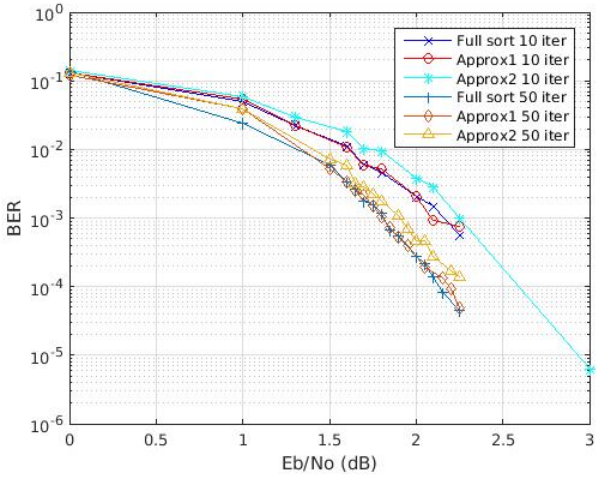

Fig. 8. Comparison of full sort with Approx1 and Approx2 for 10 iterations

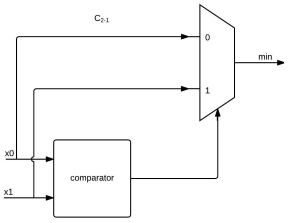

Fig. 9. Circuit for C_{2-1}

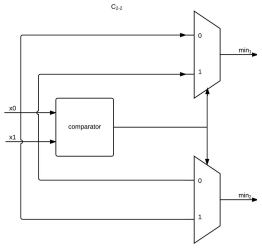

Fig. 10. Circuit for C_{2-2}

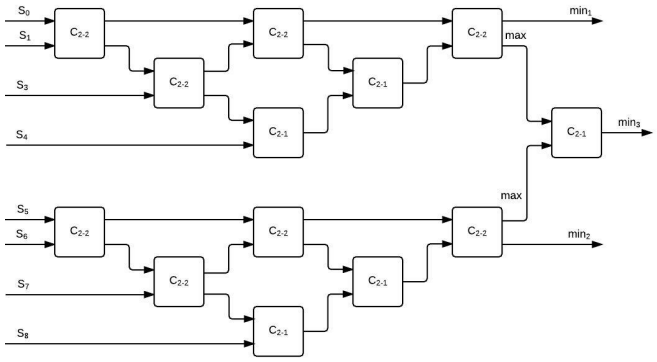

Fig. 11. Circuit that derives λ-minimum values with 13 comparisons

TABLE II
DELAY COMPARISON

	Approx1	Approx2	[2]
Delay (ns)	7.17	3.75	1.24
clock cycles	1	1	8-9

The proposed architectures are quantitatively compared in Table II in terms of latency. Furthermore, the corresponding hardware complexity is quantified in Table III. The complexities reported refer to the Virtex-6 FPGA device XC6VLX240T, speed grade -1. The proposed approximations substantially reduce hardware compared to the full parallel solution. The serial solution is of minimal complexity but it requires substantial more clock cycles (Table II).

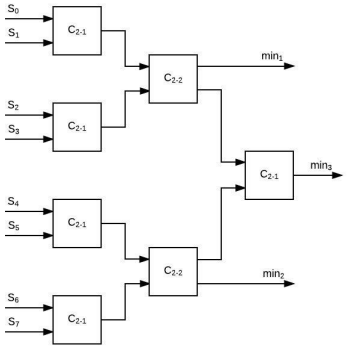

Fig. 12. Circuit that derives λ-minimum values with 7 comparisons

TABLE III
HARDWARE COMPLEXITY COMPARISON

	Approx1	Approx2	[2]
LUTs	240	126	34
Slices	108	69	22
Registers	74	73	66
Comparators	13	7	3

V. CONCLUSION

This paper proposes an approximate derivation of the λ-minimum values entering a check node during iterative decoding of LDPC codes. It has been shown that the number of comparisons required can be reduced by two times per check node with a minor impact on decoding performance, thus reducing the overall hardware complexity of the decoder. Approximate processing could be a way to further reduce the complexity of forward error correction system.

REFERENCES

[1] R. Gallager, "Low-Density Parity-Check Codes," *IRE Transactions on Information Theory*, 1962.
[2] F. Guilloud, E. Boutillon, and J.-L. D'Anger, "Decodage des codes LDPC par l'algorithme λ-min," 2003.
[3] C. Shannon, "A mathematical theory of communications," *Bell System Technical Journal, 27*, pp. 379–423, 623–656, July, October 1948.
[4] G. Battail and A. H. M. El-Sherbini, "Coding for radio channels," *Annales des Telecommunications*, vol. 7596, pp. 199–220, 1982.
[5] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Transactions on Information Theory*, vol. 42, pp. 1064–1070, 1996.
[6] M. Fossorier, M. Mihaljevic, and I. Imai, "Reduced complexity iterative decoding of low-density parity-check codes based on belief propagation," *IEEE Transactions on Communications*, vol. 47, pp. 673–680, May 1999.
[7] E. Boutillon, F. Guilloud, and J.-L. Danger, "lambda-min decoding algorithm of regular and irregular LDPC codes," in *3rd International Symposium on Turbo Codes and Related Topics*, 2003.
[8] S. W. A.-H. Baddar and K. E. Batcher, *Designing Sorting Networks: A New Paradigm*. Springer, 2011.
[9] I. Tsatsaragkos and V. Paliouras, "Approximation algorithms for identifying minima on min-sum LDPC decoders and their hardware implementation," *IEEE Transactions on Circuits and Systems - Part II*, vol. 62, no. 8, pp. 766–770, 2015.

ISBN 978-0-9928626-7-1 © EURASIP 2017