A comparison between the sum-product and the min-sum iterative detection algorithms based on density evolution

Achilleas Anastasopoulos

Electrical Engineering and Computer Science Department

University of Michigan

Ann Arbor, MI 48109-2122

***Abstract*—Recently, density evolution techniques have been used to predict the performance of iterative decoders utilizing the sum-product belief propagation algorithm. In this paper, we extend this analysis to the min-sum algorithm for binary codes. Using two representative applications, *i.e.*, low-density parity-check (LDPC) codes and repeat accumulate (RA) codes, the sum-product and min-sum algorithms are compared. The results demonstrate a performance degradation of 0.27 – 1.03 dB for the min-sum algorithm, which confirms earlier simulation results. However, it is shown in this paper, that a small modification to the min-sum algorithm results in an approximate sum-product algorithm, which performs at least as well as the original sum-product algorithm, when finite message precision is considered.**

I. Introduction

High-performance coding schemes, including turbo codes [1], low-density parity-check (LDPC) codes [2], and repeat accumulate (RA) codes [3] have been recently analyzed using density evolution [4]. For a given code ensemble, and a particular iterative detection algorithm, density evolution can be used to calculate exact expressions for the average (over the code ensemble) probability density functions (pdfs) of the messages exchanged in each iteration of the message passing algorithm. The underlying assumption is that the graph describing the particular code has a local tree structure, and that the channel and the iterative decoding algorithm satisfy some symmetry conditions [4]. Furthermore, this powerful tool can be used to predict the minimum signal to noise ratio (or maximum noise variance) for which the probability of decoding error approaches zero as the length of the code and the number of iterations approach infinity. The existence of a *concentration* theorem [4] guarantees that most of the codes in the ensemble have performance close to that predicted by density evolution.

Most of the work related to density evolution is concentrated in the sum-product [5] iterative algorithm. On the other hand, the min-sum [5] algorithm is more suited for practical implementation due to its simplicity. Simulation results [6] have demonstrated a degradation of the min-sum algorithm at low signal-to-noise (SNR) values, which seems intuitive, since the min-sum algorithm can be thought of as a high-SNR approximation to the sum-product algorithm. The absence of analytical results that confirm the simulation results, motivates the analysis and characterization of the min-sum message passing algorithm using density evolution.

In this paper, we present a comparison between the sum-product and the min-sum algorithms based on density evolution. In particular, a recursive system of equations that describe the evolution of the pdfs of the messages under the min-sum algorithm is developed. We then apply these results in both LDPC and RA codes. Furthermore, motivated by the significant performance loss observed in the min-sum algorithm, we suggest minimal modifications to the min-sum algorithm, and demonstrate significant performance gains when these modified versions are used.

II. Density evolution for the min-sum algorithm

Consider the snapshot of a factor graph [7] representing a binary code shown in Fig. 1. Under the min-sum algorithm, the

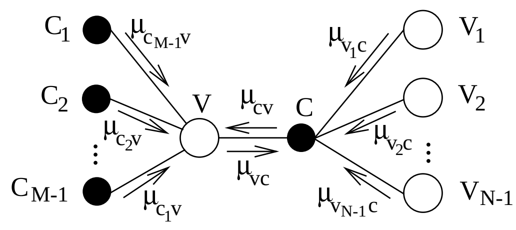

Fig. 1. Snapshot of a factor graph representing a binary code.

outgoing message from the variable node V to the check node C, denoted by μ_{VC} has the form of a log-likelihood ratio, and is given by

$$\mu_{VC} = \sum_{i=1}^{M-1} \mu_{C_iV} \tag{1a}$$

where M is the number of check nodes connected to V, and μ_{C_iV} are the corresponding incoming messages from each of the neighboring check nodes $C_i \neq C$ to the variable node V. Similarly, assuming that the check node C is performing an even parity-check, the outgoing message from the check node C to the variable node V, denoted by μ_{CV} is given by

$$\mu_{CV} = \text{sgn}(\prod_{j=1}^{N-1} \mu_{V_jC}) \min_{j\in\{1,\cdots,N-1\}} |\mu_{V_jC}| \tag{1b}$$

where N is the number of variable nodes connected to C, μ_{V_jC} are the corresponding incoming messages from each of the neighboring variable nodes $V_j \neq V$ to the check node C, and $\text{sgn}(\cdot)$ is the sign function.

Let $f^{(l)}_{C_iV}$ be the pdf of the message μ_{C_iV} at the lth iteration of the min-sum algorithm. The pdf of the outgoing message μ_{VC} at the $l+1$st iteration, denoted by $f^{(l+1)}_{VC}$, is given, according to (1a), by

$$f^{(l+1)}_{VC} = \bigotimes_{i=1}^{M-1} f^{(l)}_{C_iV} \tag{2}$$

where $\otimes$ denotes convolution. The above expression is valid as long as the messages μ_{C_iV} at the lth iteration are independent.

0-7803-7206-9/01/$17.00 © 2001 IEEE

This is true if the graph has a local tree structure [4], which, for the codes we are interested in, is guaranteed for all iterations l, as long as the interleaver size approaches infinity.

The derivation of a similar recursion for the pdf of the message μ_{CV} in (1b) is more complicated. However, since the operator described in (1b) is associative, we can first consider the case of $N-1=2$, and then generalize the result. For notational simplicity, we consider the following expression

$$C = \mathrm{sgn}(AB)\min\{|A|,|B|\} \tag{3}$$

Let f_A, F_A, f_B, F_B be the pdf and cumulative density function (cdf) of A and B, respectively. The pdf of C can be obtained as

$$\begin{aligned} f_C(x) &= f_A(x)(1-F_B(x)) + f_B(x)(1-F_A(x)) + \\ &\quad f_A(-x)F_B(-x) + f_B(-x)F_A(-x) \quad ,x>0 \quad \text{(4a)} \\ f_C(x) &= f_A(x)(1-F_B(-x)) + f_B(x)(1-F_A(-x)) + \\ &\quad f_A(-x)F_B(x) + f_B(-x)F_A(x) \quad ,x<0 \quad \text{(4b)} \end{aligned}$$

Equation (4) provides an expression for the pdf of C as a function of the pdfs of A and B, which is summarized in the following shorthand notation

$$f_C = \mathcal{F}(f_A, f_B) \tag{5}$$

Denoting the pdf of the message μ_{V_jC} at the $l+1$st iteration by $f^{(l+1)}_{V_jC}$, we can write

$$f^{(l+1)}_{CV} = \mathcal{F}(f^{(l+1)}_{V_{N-1}C}, \cdots \mathcal{F}(f^{(l+1)}_{V_3C}, \mathcal{F}(f^{(l+1)}_{V_2C}, f^{(l+1)}_{V_1C}))\cdots) \tag{6}$$

Equations (2) and (6) constitute a recursive system of equations that can be used to predict the performance of the min-sum algorithm as the number of iterations and the size of the code approaches infinity.

III. Applications

We will now apply the results of the previous section to two special cases, *i.e.*, the LDPC codes and the RA codes. Numerical evaluation of the pdfs in (2),(6) can be approached in at least three ways:

- *(i)* The pdfs are sampled and equations (2),(6) are approximately evaluated [4].
- *(ii)* The pdfs are approximated by a pdf family parametrized by a single parameter, and the infinite-dimensional recursive system of (2),(6) is transformed to a scalar nonlinear system, which can be studied analytically, or numerically [8].
- *(iii)* The original messages are quantized and the probability mass functions (pmfs) are evaluated exactly [9].

In the following section we will utilize the later method, called discretized density evolution, to compute thresholds for iterative detection receivers with quantized min-sum algorithms. The preference to method (iii) over (i) is simplicity in evaluating the pmfs, as well as the fact that method (iii) predicts the exact performance of a practical receiver that operates on quantized messages.

We would like to mention at this point that the *symmetry* condition suggested in [10], is not preserved by the transformation defined in (3), as can be verified by direct substitution in (4). The implication is that even under the Gaussian approximation [8] for the pdfs, there are two parameters (mean and variance) that need to be updated, which significantly complicates the analysis based on method (ii). Clearly, the derivation of some *symmetry* condition that is preserved under the operator defined in (3), will enable the analysis of the min-sum algorithm under the Gaussian assumption, and is an interesting future research direction.

A. Low Density Parity Check codes

The factor graph of a regular LDPC code is shown in Fig. 2, where each variable node is connected with d_v parity-check nodes and each parity-check node is connected with d_c variable nodes. The observed signal is given by

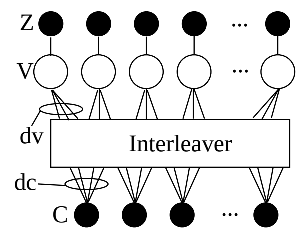

Fig. 2. Factor graph representing a regular LDPC code ($(d_v, d_c) = (3,4)$ in this example).

$$Z_k = (-1)^{V_k} + N_k \tag{7}$$

where the noise variable N_k is Gaussian with zero mean and variance σ^2. Assuming that the all-0 codeword is transmitted, the message μ_{ZV} is given by

$$\mu_{ZV} = \ln\frac{P(Z|V=0)}{P(Z|V=1)} = \frac{2Z}{\sigma^2} \tag{8}$$

with a pdf given by

$$f_{ZV}(x) = N(x; 2/\sigma^2; 4/\sigma^2) \tag{9}$$

where $N(\cdot; m; s^2)$ denotes the Gaussian pdf with mean m and variance s^2. Equations (2),(6) can be simplified, since all relevant pdfs are equal, resulting in the following recursion

$$f^{(0)}_{CV}(x) = \delta(x) \tag{10a}$$

$$f^{(l+1)}_{VC} = \left[\bigotimes_{i=1}^{d_v-1} f^{(l)}_{CV}\right] \otimes f_{ZV} \tag{10b}$$

$$\begin{aligned} f^{(l+1)}_{CV} &= \mathcal{F}(f^{(l+1)}_{VC}, \cdots \mathcal{F}(f^{(l+1)}_{VC}, \mathcal{F}(f^{(l+1)}_{VC}, f^{(l+1)}_{VC}))\cdots) \\ &\triangleq \mathcal{F}^{d_c-1} f^{(l+1)}_{VC} \end{aligned} \tag{10c}$$

with $\delta(x)$ being the Dirac delta function. The probability of bit error at the $l+1$st iteration can be evaluated as

$$P_e^{(l+1)} = \int_{-\infty}^{0} \left[f^{(l+1)}_{VC} \otimes f^{(l)}_{CV}\right](x)dx \tag{11}$$

For the implementation of discretized density evolution, an R-bit uniform quantizer was assumed, with step size $\Delta = 40/2^R$. For a given set of (d_v, d_c) values, a binary search was performed to find the maximum noise variance that results in BER less than 10^{-30} in at most 1000 iterations. The corresponding results are shown in Table I (these values are accurate up to 0.001dB). From the numerical results in Table I we observe that the thresholds corresponding to the sum-product algorithm are slightly smaller than those reported in [8]. This is due to the particular discretization considered herein. Comparing the thresholds for the sum-product and the min-sum algorithm a difference as small as 0.27dB and as large as 0.98dB is observed. The inferior performance of the min-sum algorithm can be attributed to both, the suboptimality of the min-sum algorithm, as well as the specific quantization scheme used (*i.e.*, $R = 6$ bits and $\Delta = 40/64$).

TABLE I

THRESHOLDS FOR LDPC CODES WITH SUM-PRODUCT AND MIN-SUM ALGORITHMS ($R = 6$ BITS).

d_v	d_c	rate	$\sigma_{\text{sum-prod}}$	$\sigma_{\text{min-sum}}$	diff. (dB)
3	6	0.5	0.8689	0.8177	0.53
4	8	0.5	0.8187	0.7455	0.81
5	10	0.5	0.7685	0.6957	0.86
3	5	0.4	0.9916	0.9154	0.69
4	6	1/3	0.9757	0.8716	0.98
3	4	0.25	1.2310	1.1020	0.96
4	10	0.6	0.7346	0.6776	0.70
3	9	2/3	0.7017	0.6751	0.34
3	12	0.75	0.6271	0.6079	0.27

In order to quantify the performance loss due to discretization, several values for R were examined. Specifically, the results for the LDPC code with $(d_v, d_c) = (3, 6)$ are shown in Table II. It can be observed that low resolution affects equally the sum-product and min-sum algorithms, while the advantage of the sum-product algorithm saturates at about 8 bits.[1] We would like to emphasize that no attempt was made to optimize the quantizer in these experiments, *i.e.*, a uniform quantizer is used and the quantization step $\Delta = 40/2^R$ is chosen based on trial and error tests, for the $(d_v, d_c) = (3, 6)$ LDPC code with $R = 6$.

TABLE II

THRESHOLDS FOR THE $(d_v, d_c) = (3, 6)$ LDPC CODE WITH SUM-PRODUCT AND MIN-SUM ALGORITHMS, AND DIFFERENT VALUES FOR R.

R	$\sigma_{\text{sum-prod}}$	$\sigma_{\text{min-sum}}$	diff. (dB)
3	0.6492	0.6492	0.00
4	0.7646	0.7646	0.00
5	0.8488	0.8051	0.46
6	0.8689	0.8177	0.53
7	0.8777	0.8211	0.58
8	0.8802	0.8219	0.60
9	0.8806	0.8222	0.60

[1] the threshold value for the sum-product algorithm reported in [8] is $\sigma_{\text{sum-product}} = 0.8809$

The significant advantage of the sum-product over the min-sum algorithm, which is evident from the results in Tables I, and II motivates the use of modifications to the min-sum algorithm such that the sum-product algorithm is approximated. Since the min-sum algorithm can be thought of as an approximation to the sum-product algorithm for high SNR, we seek more accurate approximations that are valid even for low SNR values. The exact sum-product algorithm is defined by a set of equations, similar to (1a),(1b). Specifically, (1a) is identical in both algorithms, while (1b) should be modified as follows

$$\mu_{CV} = 2\tanh^{-1}\left[\prod_{j=1}^{d_c-1}\tanh(\frac{\mu_{V_jC}}{2})\right] \tag{12}$$

Considering the case of two variables only for illustration, the following holds

$$\begin{aligned} C &= 2\tanh^{-1}\left[\tanh(\frac{A}{2})\tanh(\frac{B}{2})\right] \\ &= \text{sgn}(AB)\min\{|A|,|B|\} + \ln(\frac{1+e^{-|A+B|}}{1+e^{-|A-B|}}) \end{aligned} \tag{13}$$

A simple approximation to the *correction* term is given by

$$\ln(\frac{1+e^{-|A+B|}}{1+e^{-|A-B|}}) \approx \begin{cases} 0.5 & ,|A+B| \le 1, |A-B| > 1 \\ -0.5 & ,|A+B| > 1, |A-B| \le 1 \\ 0 & ,\text{else} \end{cases} \tag{14}$$

The results of implementing discretized density evolution to the LDPC codes of Table I, utilizing the message passing rule implied by (13) with the approximation in (14), are presented in Table III. These results indicate that the simple approximation to the *correction* term in (14) is sufficient, not only to close the gap between the min-sum and the sum-product algorithms, but also, to provide a slight advantage of the modified min-sum over the original sum-product algorithm. The later result might seem counterintuitive, however we emphasize that the discretized message passing algorithms are approximations to

TABLE III

THRESHOLDS FOR LDPC CODES WITH SUM-PRODUCT AND MODIFIED MIN-SUM ALGORITHMS ($R = 6$ BITS).

d_v	d_c	$\sigma_{\text{sum-prod}}$	$\sigma_{\text{mod-min-sum}}$	diff. (dB)
3	6	0.8689	0.8700	-0.01
4	8	0.8187	0.8222	-0.04
5	10	0.7685	0.7748	-0.07
3	5	0.9916	0.9934	-0.02
4	6	0.9757	0.9841	-0.07
3	4	1.2310	1.2370	-0.04
4	10	0.7346	0.7367	-0.02
3	9	0.7017	0.7021	0.00
3	12	0.6271	0.6273	0.00

the sum-product and min-sum algorithms. In this sense, there is no reason why discretized sum-product message passing should perform better than the modified min-sum, or any other ad-hoc message passing algorithm.

To verify the differences between the three iterative detection algorithms considered above, a length 4000, $(d_v, d_c) = (3, 6)$, LDPC code was simulated. A 6-bit uniform quantizer was used and a maximum of 20 iterations were considered at the receiver. The resulting BER and codeword error rate (CER) curves are shown in Fig. 3. The results show that the difference between

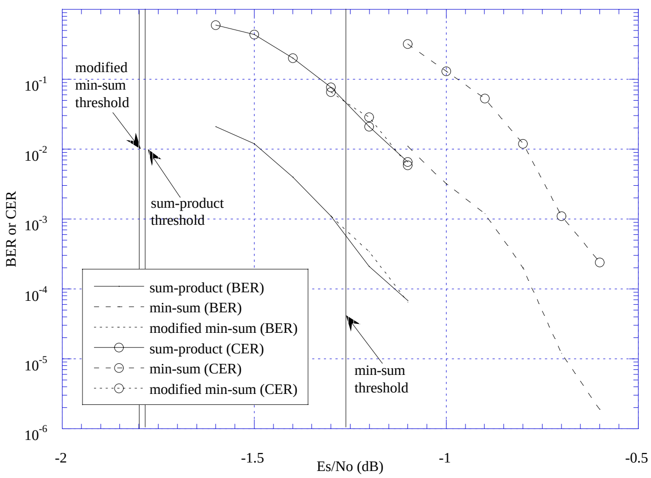

Fig. 3. BER and CER vs. $E_s/N_0 = 1/(2\sigma^2)$ for a length 4000, $(d_v, d_c) = (3, 6)$ LDPC code.

the sum-product and the min-sum algorithm is approximately 0.3dB, which is close to the value predicted using density evolution. The modified min-sum algorithm performance is almost identical to the sum-product, a fact which is also predicted using density evolution. We note that due to the small codeword length, the performance of all algorithms is far worse than the performance predicted by density evolution.

B. *Repeat Accumulate codes*

The factor graph of a regular RA code of rate $1/q$ is shown in Fig. 4. The pdf of the message μ_{ZX} is precisely the one in (9).

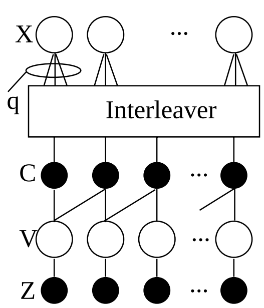

Fig. 4. Factor graph representing a regular RA code.

The message passing algorithm is slightly more complicated for RA codes, resulting in the following set of recursive updates for the message pdfs

$$f_{CV}^{(0)}(x) = \delta(x) \tag{15a}$$
$$f_{VC}^{(l+1)} = f_{CV}^{(l)} \otimes f_{ZV} \tag{15b}$$
$$f_{CX}^{(l+1)} = \mathcal{F}(f_{VC}^{(l+1)}, f_{VC}^{(l+1)}) \tag{15c}$$
$$f_{XC}^{(l+1)} = \bigotimes_{i=1}^{q-1} f_{CX}^{(l+1)} \tag{15d}$$
$$f_{CV}^{(l+1)} = \mathcal{F}(f_{VC}^{(l+1)}, f_{XC}^{(l+1)}) \tag{15e}$$

The probability of bit error at the $l+1$st iteration can be evaluated as

$$P_e^{(l+1)} = \int_{-\infty}^{0} \left[f_{XC}^{(l+1)} \otimes f_{CX}^{(l+1)} \right](x)dx \tag{16}$$

Applying discretized density evolution, we obtain the results shown in Table IV. The conclusions are similar to those for

TABLE IV

THRESHOLDS FOR RA CODES WITH SUM-PRODUCT AND MIN-SUM ALGORITHMS ($R = 6$ BITS).

q	$\sigma_{\text{sum-prod}}$	$\sigma_{\text{min-sum}}$	diff. (dB)
3	1.143	1.106	0.29
4	1.367	1.278	0.58
5	1.527	1.392	0.80
6	1.652	1.486	0.92
7	1.754	1.566	0.98
8	1.840	1.635	1.03

LDPC codes: sum-product outperforms min-sum by 0.29–1.03 dB, and the performance loss increases with q. The performance of the modified min-sum algorithm is shown in Table V, where it is verified that the modification in (14) results in a slight performance gain over the the original discretized sum-product algorithm.

TABLE V

THRESHOLDS FOR RA CODES WITH SUM-PRODUCT AND MODIFIED MIN-SUM ALGORITHMS ($R = 6$ BITS).

q	$\sigma_{\text{sum-prod}}$	$\sigma_{\text{mod-min-sum}}$	diff. (dB)
3	1.143	1.144	-0.01
4	1.367	1.369	-0.01
5	1.527	1.530	-0.02
6	1.652	1.658	-0.03
7	1.754	1.764	-0.05
8	1.840	1.855	-0.07

IV. CONCLUSIONS

In this paper, the min-sum algorithm is analyzed using density evolution. The performance of LDPC and RA codes is compared for both the discretized sum-product and the discretized min-sum algorithms. The results indicate a significant performance loss for the min-sum algorithm. An interesting and counterintuitive finding is that a small modification to the min-sum algorithm results in a significant improvement, to the point that this modified min-sum algorithm performs at least as well as the discretized sum-product algorithm.

At this point a complete characterization of the min-sum algorithm is not available. For instance, *symmetry* conditions that are preserved under the operators defined in (2) and (6), have not been investigated, while the analysis of the min-sum algorithm under the Gaussian approximation for the message pdfs is an interesting future research direction.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo-codes," in *Proc. International Conf. Communications*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

[2] R. G. Gallager, "Low density parity check codes," *IEEE Trans. Information Theory*, vol. 8, pp. 21–28, Jan. 1962.

[3] D. Divsalar, H. Jin, and R. J. McEliece, "Coding theorems for turbo-like codes," in *Proc. Allerton Conf. Commun., Control, Comp.*, (Allerton House, Illinois), pp. 201–210, 1998.

[4] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Information Theory*, vol. 47, pp. 599–618, Feb. 2001.

[5] N. Wiberg, *Codes and Decoding on General Graphs*. PhD thesis, Linköping University, Linköping, Sweden, 1996.

[6] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Soft-output decoding algorithms for continuous decoding of parallel concatenated convolutional codes," in *Proc. International Conf. Communications*, (Dallas, Texas), pp. 112–117, June 1996.

[7] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Information Theory*, vol. 47, pp. 498–519, Feb. 2001.

[8] S.-Y. Chung, T. J. Richardson, and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Information Theory*, vol. 47, pp. 657–670, Feb. 2001.

[9] S.-Y. Chung, G. D. Forney, Jr., T. J. Richardson, and R. L. Urbanke, "On the design of of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, pp. 58–60, Feb. 2001.

[10] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Information Theory*, vol. 47, pp. 619–637, Feb. 2001.