Stochastic Decoding of LDPC Codes

Saeed Sharifi Tehrani, *Student Member, IEEE*, Warren J. Gross, *Member, IEEE*, and Shie Mannor, *Member, IEEE*

***Abstract*— This letter presents the first successful method for iterative stochastic decoding of state-of-the-art Low-Density Parity-Check (LDPC) codes. The proposed method shows the viability of the stochastic approach for decoding LDPC codes on factor graphs. In addition, simulation results for a 200 and a 1024 length LDPC code demonstrate the near-optimal performance of this method with respect to sum-product decoding. The proposed method has a significant potential for high-throughput and/or low complexity iterative decoding.**

***Index Terms*— LDPC codes, iterative stochastic decoding.**

I. Introduction

THE class of LDPC codes has been shown to include some of the most powerful known capacity-approaching codes [1]. LDPC codes are decoded by means of iterative belief propagation using the Sum-Product (SP) algorithm. The SP algorithm involves passing messages over the edges of a bipartite *factor graph* [2]. The implementation of LDPC decoders tends to be complex and solutions for less complex decoders are a focal point of research.

Stochastic computation was introduced in the 1960's as a method to design low-precision digital circuits [3]. This method has been used to implement neural networks [4] and recently considered for iterative decoding of some specific error-correcting codes [5]–[9]. The main advantage of this method is that probabilities can be manipulated using very simple circuits. This feature is especially interesting for the implementation of LDPC decoders in which hardware complexity and routing congestion are major problems.

Up to the present, stochastic decoding methods were successful for either short or some specific error-correcting codes. In [5] and [6] stochastic decoders are used for decoding a (7,4) Hamming and a (16,8) LDPC code. A new form of stochastic algorithm is proposed in [7] and used for trellis decoding of an acyclic (16,11) Hamming code and a (256,121) product Turbo code based on 32 component decoders of this Hamming code. CMOS and gate level circuits, and some modifications for this algorithm are discussed in [8]. Finally, the implementation of a stochastic decoder for a special acyclic (16,8) LDPC code is described in [9]. To the best of our knowledge, none of the above mentioned methods could successfully decode state-of-the-art capacity-approaching LDPC codes on factor graphs. In this letter, we propose a method for stochastic decoding of LDPC codes. Compared to floating point SP decoding, this method provides comparable performance for a (200,100) and near-optimal performance for a (1024,512) LDPC code.

Manuscript received April 14, 2006. The associate editor coordinating the review of this letter and approving it for publication was Dr. Vladimir Stankovic.

The authors are with the Dept. of Electrical and Computer Engineering, McGill University, Montreal, QC, H3A 2A7 Canada (email: {sshari9, wjgross, shie}@ece.mcgill.ca).

Digital Object Identifier 10.1109/LCOMM.2006.060570.

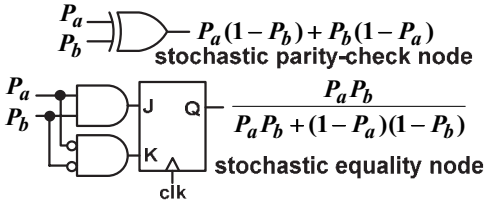

Fig. 1. Parity-check and equality nodes in stochastic decoding.

Both codes are regular, with no length-4 or 6 cycles and, have degree 3 equality nodes ($d_e = 3$) and degree 6 check nodes ($d_c = 6$).

II. Stochastic Computation on Factor Graphs

In stochastic decoding, probabilities are encoded by a Bernoulli sequence as a random sequence of digital bits in which each bit of the sequence is equal to '1' with the probability to be encoded. As an example, a sequence of N bits with m bits equal to '1', represents the probability of m/N. The encoding scheme is not unique and different encoded stochastic sequences are possible for the same probability. The stochastic representation of messages in the code factor graph results in low complexity parity-check and equality nodes. Let $P_a = \Pr(a = 1)$ and $P_b = \Pr(b = 1)$ be the probability of two input bits (a and b) in a $d_c = 3$ check node. The output probability P_c can be computed as

$$P_c = P_a(1 - P_b) + P_b(1 - P_a). \tag{1}$$

Similarly, the equality function in a $d_e = 3$ equality node is

$$P_c = P_aP_b/[P_aP_b + (1 - P_a)(1 - P_b)]. \tag{2}$$

Fig. 1 shows the equivalent hardware structures for (1) and (2). Higher degree parity-check or equality nodes can be easily converted to sub-graphs containing only degree three parity-check or equality nodes [6]. Note that the equality node in Fig. 1 *holds* on the previous output bit on the edge, if the corresponding input bits of the node are not equal.

In addition to low complexity, stochastic computation reduces the routing congestion in the decoder, because only one bit (in each direction) is needed to represent each edge between parity-check and equality nodes. Therefore, the decoding algorithm proceeds by the nodes exchanging bits along each edge in the graph. We refer to decoding rounds as Decoding Cycles (DCs) to emphasize that they do not correspond directly to the iterations in SP decoding.

III. Stochastic Decoding of LDPC Codes

One major difficulty observed in stochastic decoding is the sensitivity to the level of random switching activity (bit transition) for proper decoding operation [8]. The problem of

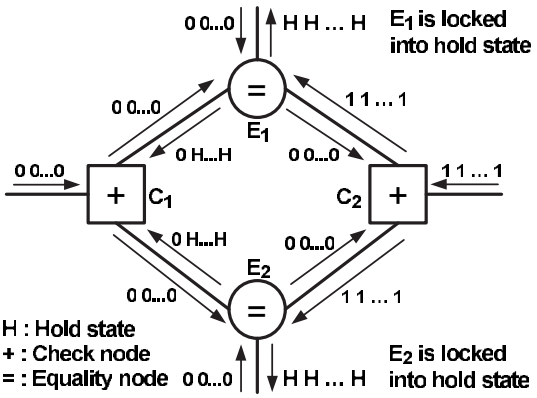

Fig. 2. An example of latching within a 4-cycle in a factor graph.

latching is described in [7] for stochastic decoding on graphs with cycles. The latching problem refers to the case where a cycle in the code graph causes a group of nodes to lock into a fixed state which is solely maintained by the messages within the cycle [7]. The latching problem is particulary acute in LDPC decoders [8]. Fig. 2 illustrates how the lack of switching activity within a 4-cycle can cause equality nodes to lock into a fixed state ("hold" state in this example). Note that latching can be worse at high SNRs in which the received Log-Likelihood Ratios (LLRs) become so large so that the corresponding probabilities approach 0 (or 1). In this case, bits in stochastic sequences are mostly '0' (or '1'), hence random switching events become too rare for proper decoding [8].

In [7], the idea of a *packetized supernode* is proposed to avoid latching by preventing correlation between messages. A supernode is a special equality node which tabulates the incoming stochastic messages in histograms, estimates their probability and regenerates uncorrelated stochastic messages by random number generators. Supernodes were used in [7] for trellis decoding of a (256,121) product Turbo code. All the equality nodes were supernodes and they were packetized in a sense that they were invoking the conventional SP calculation after a time-step to calculate the probabilities of the new outgoing messages. In addition to supernodes, the idea of scaling channel LLRs is suggested in [8] and used in stochastic decoding of a (16,11) Hamming code. In this method, every block of the received LLRs are scaled to a maximum value to ensure the same level of switching activity for each block.

Our approach uses a scaling method which increases the level of switching activity over different ranges of SNRs. In addition, we propose a method to avoid latching. This method relies on the idea of stochastic decoding and does not use the conventional SP calculation and packetized bit sequences as in supernodes. Both methods are essential for decoding relatively long LDPC codes.

A. Noise-Dependant Scaling

In Noise-Dependant Scaling (NDS), the received channel LLRs are down-scaled by a scaling factor which is proportional to the SNR. The down-scaled LLRs result in probabilities which introduce more switching activity in the stochastic decoder. Because the scaling factor is proportional to the noise level, it ensures a similar level of switching activity for different SNRs. Assuming a BPSK (± 1) transmission over an additive white Gaussian noise channel, the scaled LLR (L'_i) for the i-th symbol (y_i) in the received block is calculated as

$$L'_i = (\frac{\alpha N_0}{Y})L_i = \frac{4\alpha y_i}{Y}, \tag{3}$$

where N_0 is the single-sided noise power spectral density, $L_i = 4y_i/N_0$ is the channel LLR for y_i, Y is a fixed maximum value of the received symbols and, α is a constant factor $0 < \alpha < Y$. For BPSK transmission, we used $Y = 6$ and, for the codes we used, an α around 3 resulted in the best Bit Error Rate (BER) performance. The NDS method is often sufficient for decoding short error-correcting codes (see Fig. 3). However, for longer codes, further improvement is needed.

B. Edge Memories

To increase the random switching activity of stochastic messages in the code graph especially during latching, an M-bit shift register is introduced at each edge in the graph. We refer to these registers as Edge Memories (EMs). Each EM is updated in accordance with (2) only when the equality node is not in hold state for that edge. In case of a hold state, a bit is randomly chosen from the corresponding EM and passed through the edge as the outgoing bit. This updating scheme reduces the chance of locking into a fixed state since every time a hold state happens, a bit is randomly chosen from those previous outputs which are not produced in hold states.

C. The Proposed Stochastic Decoding Method

The proposed decoding method exploits both NDS and EMs. Upon receiving each block, the channel LLRs are scaled as in (3). In each DC, one bit of corresponding stochastic messages is passed to each equality node. If a hold state occurs on an edge, the equality node randomly picks a bit from the corresponding EM. Otherwise, the output bit of the equality node is passed through the graph and EM is updated. The check nodes perform the parity-check equation and pass their messages to the equality nodes. Each equality node calculates its output at the end of each DC and passes it to an up/down counter. The counter is decremented in case of a '0' output and incremented in case of a '1'. At any given DC, the sign bit of the counter indicates the hard decision, with a '0' sign bit indicating a decoded '+1' and a '1' sign bit indicating a decoded '-1'. The decoder operates for a fixed number of DCs unless all the parity-check equations are satisfied sooner.

IV. Simulation Results

Fig. 3 shows the BER performance for stochastic decoding of a (7,4) Hamming code using NDS with $\alpha = 3$ and maximum 2K DCs (EMs are not used). As shown, NDS improves the performance of stochastic decoding and provides comparable performance for different SNRs with respect to SP decoding. These performance results are superior than the results in [5].

Figs. 4 and 5 show the BER performance of the proposed stochastic decoding method for (200,100) and (1024,512) LDPC codes. We used $M = 25$ and maximum 10K DCs for decoding the (200,100) code and, $M = 50$ and maximum 60K DCs for the (1024,512) code. An $\alpha = 3$ is used for both

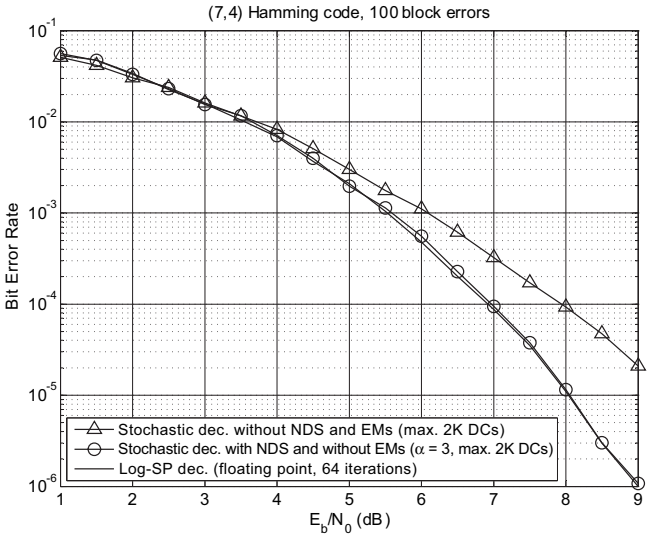

Fig. 3. Simulation results for a (7,4) Hamming code using NDS method.

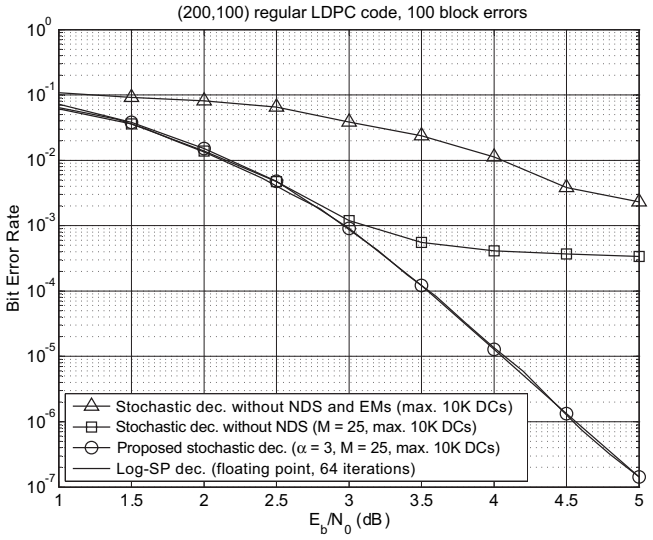

Fig. 4. Simulation results for (200,100) LDPC code.

codes. As shown, with respect to floating point SP decoding with 64 iterations[1], the proposed method provides comparable BER performance for the (200,100) code and near-optimal performance for the (1024,512) code. An SNR loss of about 0.1 dB is observed for the latter code at the BER of 10^{-6}. The observed average DCs per block are much less than the maximum DCs. The average DCs for the (200,100) code is about 200 DCs at the BER of 10^{-7} and for the (1024,512) code, it is about 5K DCs at the BER of 10^{-6}. Note that DCs are not equivalent to the iterations in SP decoding and due to the low complexity of a stochastic decoder, the clock rate can be much higher than that of a fixed point SP decoder. Also, it is more viable to have a fully-parallel stochastic decoder.

To show the performance contribution of NDS and EMs, results for (i) decoding without NDS and EMs and, (ii) decoding with EMs but without NDS are also depicted in Figs. 4 and 5. The contribution of EMs can be observed by comparing results for case (i) and (ii). Also, the contribution of NDS at higher SNRs can be easily seen by comparing the results of the proposed method and case (ii). It is worth mentioning that unlike the results of the short Hamming code in Fig. 3, NDS cannot improve the poor decoding performance for the case where EMs are not used.

V. Conclusions

A method for iterative stochastic decoding of LDPC codes is proposed. Compared to floating point SP decoding, results show comparable performance for a (200,100) and near-optimal performance for a (1024,512) LDPC code. For the first time, these results indicate the viability of the stochastic approach for decoding state-of-the-art LDPC codes on factor graphs. They also open a broad set of research questions with significant potential for high-throughput and/or low complexity iterative stochastic decoding of error-correcting codes.

Acknowledgement

The authors would like to thank V. Gaudet, S. Howard and A. Rapley for helpful discussions.

[1]No major BER improvement is observed after the 64th iteration.

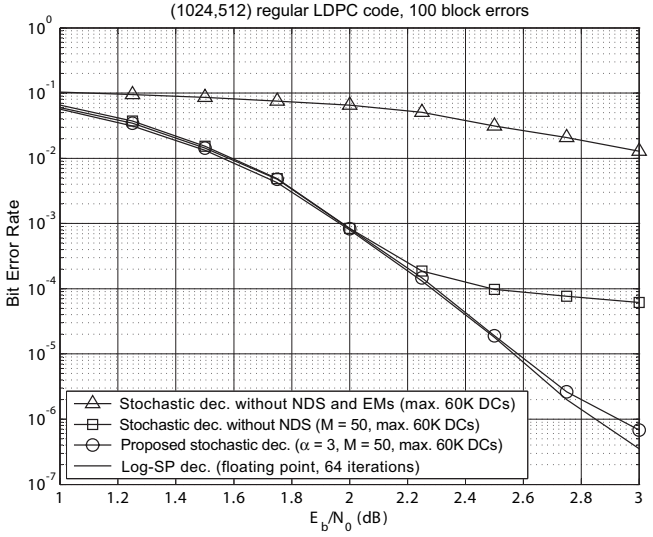

Fig. 5. Simulation results for (1024,512) LDPC code.

References

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

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

[3] B. Gaines, *Advances in Information Systems Science*. New York: Plenum, 1969, ch. 2, pp. 37–172.

[4] B. Brown and H. Card, "Stochastic neural computation I: computational elements," *IEEE Trans. Comput.*, vol. 50, pp. 891–905, Sept. 2001.

[5] A. Rapley, C. Winstead, V. Gaudet, and C. Schlegel, "Stochastic iterative decoding on factor graphs," in *Proc. 3rd Int. Symp. on Turbo Codes and Related Topics 2003*, pp. 507–510.

[6] V. Gaudet and A. Rapley, "Iterative decoding using stochastic computation," *Electronics Lett.*, vol. 39, pp. 299–301, Feb. 2003.

[7] C. Winstead, V. Gaudet, A. Rapley, and C. Schlegel, "Stochastic iterative decoders," in *Proc. IEEE Int. Symp. on Information Theory 2005*, pp. 1116–1120.

[8] C. Winstead, "Error-control decoders and probabilistic computation," in *Proc. Tohoku Univ. 3rd SOIM-COE Conf.*, Oct. 2005.

[9] W. Gross, V. Gaudet, and A. Milner, "Stochastic implementation of LDPC decoders," in *Proc. 39th Asilomar Conf. on Signals, Systems, and Computers*, Nov. 2005.