HAL
open science

Stochastic Decoding of Linear Block Codes With High-Density Parity-Check Matrices

S. Sharifi Tehrani, Christophe Jego, Bo Zhu, W.J. Gross

To cite this version:

S. Sharifi Tehrani, Christophe Jego, Bo Zhu, W.J. Gross. Stochastic Decoding of Linear Block Codes With High-Density Parity-Check Matrices. IEEE Transactions on Signal Processing, 2008, 56 (11), pp.5733 -5739. 10.1109/TSP.2008.929337 . hal-00538600

HAL Id: hal-00538600

https://hal.science/hal-00538600v1

Submitted on 22 Nov 2010

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.

Stochastic Decoding of Linear Block Codes with High-Density Parity-Check Matrices

Saeed Sharifi Tehrani[†], *Student Member, IEEE*, Christophe Jego[‡], *Member, IEEE*, Bo Zhu[†], and Warren J. Gross[†], *Member, IEEE*

***Abstract*— Stochastic decoding is a new alternative approach for iterative decoding on graphs. The capability of stochastic decoding has been recently validated for decoding Low-Density Parity-Check (LDPC) codes. This article extends the application of the stochastic approach to decode linear block codes with high-density Parity-Check Matrices (PCMs) such as Bose-Chaudhuri-Hocquenghem (BCH) codes, Reed-Solomon (RS) codes and, block turbo codes based on BCH component codes. We show how the stochastic approach, despise its bit-serial nature, is able to generate soft-output information for iterative Soft-Input Soft-Output (SISO) decoding. This article describes the structure of high-degree stochastic variable nodes used in codes with high-density PCMs. Simulation results for a $(128, 120)$ BCH code, a $(31, 25)$ and a $(63, 55)$ RS code and, a $(256, 121)$ and a $(1024, 676)$ BCH block turbo code demonstrate decoding performance close to the iterative SISO decoding with floating-point implementation.**

I. Introduction

Iterative Soft-Input Soft-Output (SISO) decoding was first introduced in 1993 for the turbo decoding of concatenated convolutional codes [1] to provide performance approaching the Shannon capacity limit. Since then, the concept of SISO decoding has been extended for decoding different classes of error-correcting codes such as product codes and Low-Density Partiy-Check (LDPC) codes. The well known Belief Propagation (BP) algorithm is also an iterative SISO-based algorithm for decoding linear block codes. This algorithm is the main algorithm for LDPC decoding. In BP, soft information, in the form of probability messages, propagates between computing nodes connected by edges of a *factor graph* [2] which is defined by the Parity-Check Matrix (PCM) of the code. The propagation of soft information in BP can be performed in parallel. This inherent parallelism together with excellent performance of LDPC codes have made LDPC decoders attractive for recent high data rate applications in digital communication.

Stochastic decoding is a new approach for decoding error-correcting codes. This approach is inspired by stochastic computation [3] where operations on probabilities are bit-serially performed on streams (or frames in some implementations) of randomly generated stochastic bits. The two main appealing features of this approach for iterative decoding are very simple hardware structures of computing nodes and significantly reduced routing and interconnection requirements. These features result in great potential for low-complexity and/or high-throughput decoding. Early stochastic methods were only successful for decoding special short/acyclic Hamming or LDPC codes [4]–[6] and, for trellis decoding of a (256,121) block turbo code based on acyclic (16,11) Hamming component decoders [7], [8]. These methods result in poor decoding performance when used to decode practical codes on factor graphs (with cycles). A new stochastic method was recently proposed that shows that stochastic decoding is able to provide near-optimal performance for decoding practical LDPC codes with respect to the floating-point BP [9]. The potential of this method for low-complexity decoding was recently validated by an FPGA implementation of a (1024,512) LDPC decoder. This implementation provides a throughput of 706 Mbps at a Bit Error Rate (BER) of 10^{-6} with performance loss of about 0.1 dB, compared to the floating-point BP, while occupying only 36% of a Virtex-4 LX200 FPGA device [10].

Despite the inherent parallelism and excellent performance of BP for LDPC decoding, it is known that the BP algorithm is not suitable for decoding codes with non-sparse PCMs. For this reason, BP can not be directly applied for decoding many popular and powerful codes such as Bose-Chaudhuri-Hocquenghem (BCH) and Reed-Solomon (RS) codes. This problem was recently investigated in [11] and a new Adaptive Belief Propagation (ABP) was suggested for RS decoding. It was shown that when BP is applied to a code with high-density PCM, it is likely that BP gets stuck at some local minimum points with small gradient which corresponds to some unreliable symbols. Based on this observation, at each iteration of ABP, the PCM of the code is adapted according to the bit reliabilities to sparsify those columns in the PCM which are associated with unreliable bits. ABP offers a decoding gain of more than 3 dB over hard-decision RS decoding. It also benefits from the parallel message passing in BP. However, the PCM adaptation step in ABP is complex. Inspired by ABP, a novel method for the Turbo-oriented Adaptive Belief propagation (TAB) was recently proposed for turbo decoding of product codes [12]. In TAB, the PCM adaptation is performed before the BP process thus the PCM adaptation is not required during iterations of the BP process in the component decoder. This feature significantly decreases decoding complexity. In addition, the TAB outperforms the ABP and provides a performance close to the Chase-Pyndiah (CP) algorithm [13] for iterative decoding of product codes.

This article presents a method which extends the application of stochastic decoding to the important and popular classes of

Manuscript received M D, Y. The associate editor for coordinating the review of this paper ...

The authors are with the Dept. of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada[†] and GET/ENST Bretagne, CNRS TAMCIC UMR2872, 29238 Brest, France[‡] (email: wjgross@ece.mcgill.ca).

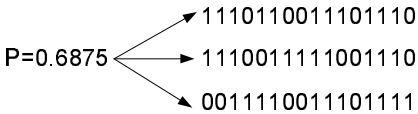

Fig. 1. Some possible stochastic streams the probability of 0.6875.

BCH, RS and block turbo codes. It shows how the stochastic approach, with its bit-serial nature, can be used with the reliability-based PCM adaptation technique used in the ABP and the TAB. In addition, the article describes the structure of high-degree stochastic Variable Nodes (VNs). To the best of our knowledge, results provided in the paper are the first results in the literature for stochastic decoding of high-density linear block codes on factor graphs. These results show decoding performance close to the floating-point ABP and/or TAB. The rest of the paper is organized as follows. Section II provides an overview of stochastic decoding, the ABP and the TAB. Section III presents the proposed decoding method. In Section V, simulation results are presented. Finally, Section VI offers the concluding remarks and the implications of the results.

II. Background

A. Stochastic Representation and Decoding

In stochastic decoding, probabilities received from the channel are transformed to streams of stochastic bits using Bernoulli sequences. Each bit in the stream is likely to be '1' with the probability to be transformed. This transformation is not one-to-one and different stochastic streams are possible for the same probability. For example, Fig. 1 shows some possible streams for a probability of 0.6875. In this figure, 11 bits out of 16 bits in each stream are '1', therefore, each stream represents a probability of 0.6875. Stochastic streams are not necessarily frames of bits and they can be interpreted and used as stochastic processes in which no framing is required [14]. Using stochastic representation, operation on probabilities are transformed to bit-serial operations on stochastic streams using simple processing elements. For instance, an AND gate can be used for multiplication of two probabilities.

The simplicity of stochastic computation is favorable for iterative graph-based decoding such as BP decoding. Stochastic representation results in simple structure of the computing nodes and more importantly, due to its bit-serial nature, it lessens the routing congestion and interconnection problem associated with the implementation of BP-based decoding algorithm such as LDPC decoders. This is because only one wire per direction is required to represent an edge in the graph. Let stochastic bit streams $\{a_i\}$ and $\{b_i\}$ represent the input probabilities of $P_a = \Pr(a_i = 1)$ and $P_b = \Pr(b_i = 1)$, respectively, and $\{c_i\}$ represent the output probability $P_c = \Pr(c_i = 1)$. Fig. 2 shows the equivalent structures for a degree-3 ($d_v = 3$) VN and a degree-3 ($d_c = 3$) Parity-check Node (PN) [4], [9], [10]. The VN shown in Fig. 2 is in the *hold state* (*i.e.*, $c_i = c_{i-1}$) when the input bits are not equal ($a_i \neq b_i$). The output bits of a VN which are not generated in the hold state are referred to as *regenerative bits* [10]. Regenerative bits

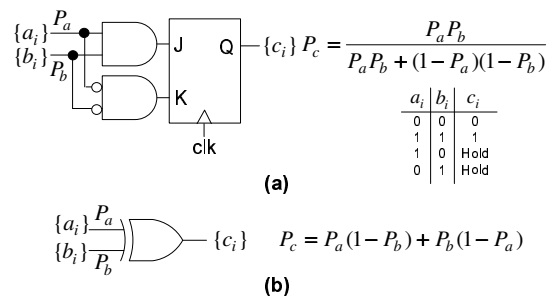

Fig. 2. (a) Stochastic variable node and (b) parity-check node [4], [9], [10].

are important for proper stochastic decoding [10]. Stochastic decoding proceeds by VNs and PNs exchanging bits along the edges in the factor graph. Each stochastic decoding round is referred to as a *Decoding Cycle* (DC). A DC refers to the exchange of one bit between VNs and PNs over the edges of factor graph and hence does not directly correspond to one iteration in BP [9].

The VN structure in Fig. 2 cannot be directly applied for decoding practical linear block codes. Stochastic decoding is sensitive to the level of switching activity (bit transitions) within the code graph, especially at high Signal-to-Noise Ratios (SNRs). The *latching* (lock-up) problem is also a major problem in stochastic decoding [7], [9]. This problem refers to the case where the existence of cycles in the code graph correlates stochastic streams in such a way that some nodes are stuck into a fixed state for several DCs. These problems were recently investigated in [9] for LDPC decoding and two methods were suggested to increase the switching activity and alleviate the latching problem:

1) Noise-Dependant Scaling (NDS): In NDS, the received channel Log-Likelihood-Ratios (LLRs), $\mathbf{L_{ch}}$, are scaled by a factor which is proportional to the noise level. This is to ensure a similar level of switching activity for different SNRs. Assuming a Binary Phase-Shift Keying (BPSK) transmission (± 1) over an Additive White Gaussian Noise (AWGN) channel with a power spectral density of N_0 , the scaled LLRs in NDS are calculated as

$$L'_i = (\frac{\alpha N_0}{Y}) L_i, \tag{1}$$

where L_i is the channel LLR for the i-th received symbol in the block, Y is a fixed maximum value for the received symbols and, α is a constant factor whose value can be chosen based on decoding performance of the decoder [9].

2) Edge Memory (EM): In addition to NDS, EMs are also essential for proper stochastic decoding [9]. EMs are memories assigned to edges of the factor graph and can be implemented as M-bit shift registers. EMs operates with regenerative bits and are used to decorrelate and rerandomize the stochastic streams and hence alleviate the latching problem in stochastic decoders [9], [14]. When a VN is not in a hold state for the edge, the corresponding EM is updated with the regenerative output bit of the node in a first-in-first-out manner. In the case of a hold state, a bit is randomly chosen from the EM as the output bit of the node and the EM is not updated [9], [10].

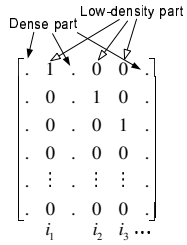

Fig. 3. Form of an adapted parity-check matrix.

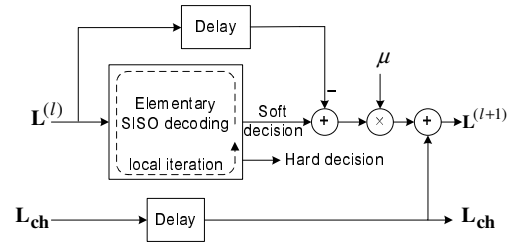

Fig. 4. Block turbo decoding.

B. Adaptive Belief Propagation

BCH and RS codes are important classes of linear cyclic error-correcting codes with multiple error detection and correction capability. For the sake of simplicity, this section briefly provides an overview of the ABP for SISO RS decoding [11] over $\mathrm{GF}(2^m)$, however, the same approach is applicable for binary BCH codes. The PCM of an (n, k) RS code over $\mathrm{GF}(2^m)$ can be represented by

$$H = \begin{bmatrix} 1 & \beta & \dots & \beta^n \\ 1 & \beta^2 & \dots & \beta^{2(n-1)} \\ \vdots & \vdots & \dots & \vdots \\ 1 & \beta^{(d-1)} & \dots & \beta^{(d-1)(n-1)} \end{bmatrix}, \quad (2)$$

where β is the primitive element in $\mathrm{GF}(2^m)$ and $d_{min} = n - k + 1$ is the minimum distance of the code. This code can correct up to $t = \lfloor (d_{min} - 1)/2 \rfloor$ symbols. In ABP, the PCM, H, is expanded to its binary representation, H_b, by substituting each codeword in the $\mathrm{GF}(2^m)$ with its equivalent binary representation [11]. This representation transforms the problem of RS decoding to the general problem of decoding of an (N, K) binary block code with $N = m \times n$ and $K = m \times k$.

Let l denote the iteration step. At $l = 0$, ABP starts with $\mathbf{L_{ch}}$ and the binary PCM, $H_b^{(0)} = H_b$. At each iteration of ABP, two steps are performed: the reliability-based adaptation of H_b and the generation of extrinsic information using BP [11]. In the adaptation step, the LLRs are sorted based on their absolute value in an ascending manner and ordering indices are stored. Let $\mathbf{L}^{(l)} = \{L_{i_1}^{(l)}, ..., L_{i_N}^{(l)}\}$ be the sorted list of LLRs and $\{i_1, ...i_N\}$ be the stored indices. The first LLR, $L_{i_1}^{(l)}$, corresponds the least reliable bit (the i_1-*th* bit in the block) and the last LLR, $L_{i_N}^{(l)}$ corresponds to the most reliable bit (the i_N-*th* bit in the block). After this phase, starting from $j = i_1$ to i_N, row operations are performed to systematize the j-*th* column of $H_b^{(l)}$ to form a unity weight column $e_j = [0..010..0]^T$, in which the only non-zero element is placed at the j-*th* position. If such systematization is not possible for a column, the algorithm proceeds to the next column in $\mathbf{L}^{(l)}$. This procedure can be done using the Gaussian elimination method. Fig. 3 shows the form of an adapted PCM which is decomposed into dense and low-density parts.

After the adaptation step, the BP is applied on the sorted LLRs, $\mathbf{L}^{(l)}$, based on the adapted PCM, $H_b^{(l)}$, to generate the extrinsic LLRs, $\mathbf{L}_{ext}^{(l)}$. The LLR $\mathbf{L}^{(l+1)}$ is then updated according to

$$\mathbf{L}^{(l+1)} = \mathbf{L}^{(l)} + \lambda \times \mathbf{L}_{ext}^{(l)}, \quad (3)$$

where $0 < \lambda < 1$ is a damping coefficient. The algorithm returns to the adaptation step unless it has been run for a fixed maximum number of iterations, l_{max}, or all the parity-checks are satisfied [11]. Several variations of the ABP are given in [11]. One of them that considers a *Hard-Decision Decoding* (HDD) step at the end of each iteration of the decoding algorithm is especially attractive. This variation consists of performing hard decisions on LLRs produced by the BP. At the end of the decoding process, the most likely codeword is selected. This variation improves the convergence and the performance of the iterative RS decoder [11].

C. Turbo-oriented Adaptive Belief Propagation

TAB is an innovative method for block turbo decoding using BP-based decoders as elementary SISO component decoders. TAB is inspired by ABP but it is less complex than ABP and offers better decoding performance. Fig. 4 shows the principles of SISO block turbo decoding. It includes sequential decoding of rows and columns of the component codes and the iterative process. The "global" iterations, l, have to be distinguished from the iterations of the BP process which we call "local" iterations. As mentioned earlier, ABP requires the adaptation of H_b and the generation of extrinsic information using BP in each local iteration. The Gaussian elimination used in the adaptation step of ABP is a computationally-expensive process. In TAB, the adaption step is only performed at beginning of each global iteration (before the BP process). This means that the PCM is the same during the BP process and no damping coefficient or matrix adaptation is necessary during local iterations. Instead, the LLR update is performed in the global iteration of turbo process using μ coefficient. This innovation significantly reduces the complexity. In addition, the TAB outperforms ABP for block turbo decoding and provides performance close to the CP algorithm. Another advantage of TAB is its high degree of parallelism for high data rate applications compared to the CP decoding [12].

III. The Proposed Method

The binary PCM of an RS or BCH code is a dense matrix which results in a factor graph with high-degree VNs and PNs (with degrees much higher than the degree of nodes in LDPC codes). For instance, the factor graph of the (63,55) RS code used in this paper, has 378 VNs and 48 PNs over $\mathrm{GF}(2^6)$. The maximum degrees of VNs and PNs are 34 and 184, respectively, and about 77% of VNs have a $d_v \geq 20$. In ABP, the complexity of high-degree nodes is high and a dense

PCM obliges complex routing requirements. For example, using 8-bit precision for probabilies in ABP, the factor graph of the RS code requires more than 127K wires to represent edges in two directions. The stochastic approach offers nodes with low complexity[1] and significantly alleviates the routing requirements. Using this approach about 16K wires are needed for the (63,55) RS code. The stochastic method used for decoding high-density linear block codes employs NDS and EMs. In addition, due to the high-degree nodes and the PCM adaptation step in the ABP and the TAB, it exploits new stochastic techniques which are discussed as follows.

A. High-Degree Stochastic Variable Nodes

The construction of high-degree stochastic PNs is straightforward and is based on XORing input bits. However, the situation for high-degree VNs is different because of the hold state. For proper decoding performance, it is essential that a high-degree stochastic VN is constructed based on subgraphs of low-degree stochastic VNs (*e.g.*, with $dv \leq 4$ subnodes). To elaborate, consider the structure of the stochastic VN in Fig. 5(a) with an arbitrary d_v. This stochastic structure is not suitable for high-degree VNs. This is because this structure is entirely in the hold state when any two input bits are not equal. Therefore, by using this structure the chance of being in a hold state increases, as d_v increases. An increased chance of hold state for VNs reduces the regenerative bits propagating in the graph and results in the less switching activity within the graph and thus, degraded stochastic decoding performance. Note that this phenomenon can be very destructive when bits in input stochastic streams of VNs are mostly '0' (or '1'), for instance, at high SNRs where corresponding probability messages are either close to 0 (or 1) or, during the convergence to the right codeword where most PNs are satisfied and only a few PNs remained unsatisfied. These problems are less likely for the stochastic structure shown in Fig. 5(b) which is constructed based on $d_v = 3$ subnodes (with 1-bit memory). This is because by having stochastic input bits which are either mostly '0' (or '1'), the chance of a hold state for the dashed exit subnode in Fig. 5(b) is much less. Therefore, the entire node is less likely to be in the hold state. Note that in Fig. 5(b), an EM is used only for the exit output edge. Fig. 6 shows an example which compares the averaged percentage of the hold state between two $d_v = 9$ VNs based on structures in Figs. 5(a) and (b) for the case where P_1 is varying and $P_2 = P_3 = ... = P_8 = 0.9$.

B. Representing Soft-Output Information

The ABP and the TAB relies on reliability-based PCM adaptation as well as the reliability update scheme in (3). Both of these steps use the soft-output information provided by BP. Since the BP-based decoders inherently operates on reliabilities, it can be easily incorporated into the adaptation and the update scheme. This situation is, however, different for stochastic decoding methods. Stochastic methods convert

[1]For example, PNs in the BP-based decoding implement multiplication and the $tanh(\cdot)$ processing (or an approximation of it), while in stochastic decoding PNs are simply XOR gates.

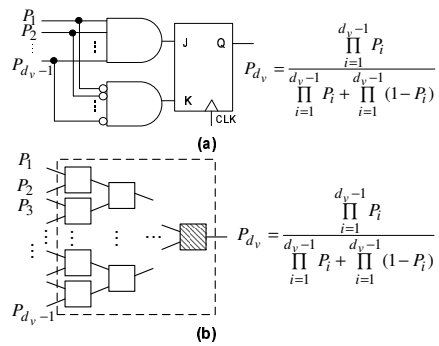

Fig. 5. (a) A structure which is not suitable for high-degree VNs. (b) An example of constructing a high-degree VN based on low-degree subnodes.

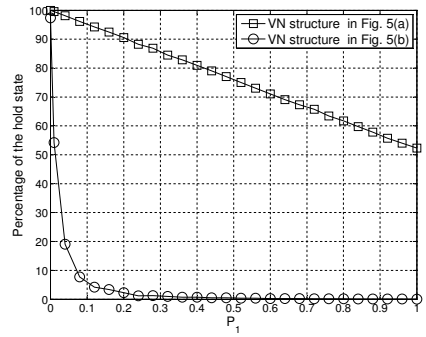

Fig. 6. Percentage of holds on the output bits of two $d_v = 9$ VNs based on structures in Fig. 5(a) and (b), P_1 is varying and $P_2 = ... = P_8 = 0.9$.

the channel reliabilities to stochastic bit streams and decoding entirely proceeds in a bit-serial fashion. To incorporate with the reliability update and the adaptation steps, it is essential that the stochastic decoding method provides soft-output information. For this purpose, we propose to use saturating up/down counters to represent soft-output information. In this technique, a counter is assigned to each VN. The counter can be initialized to contain zero value. The counter is incremented if the corresponding VN output is '1', unless the counter has reached its maximum limit ($+U$). Similarly, when the VN output is '0' the counter is decremented, unless it has reached its minimum limit ($-U$). After a number of DCs, the contents of the counters can be converted to soft information and the LLR update and the PCM adaptation steps can be performed.

Let V be the value of a saturating up/down counter associated with a VN ($-U \leq V \leq +U$). This value can be transformed to soft-information in the probability domain as $P_{ext} = (V + U)/2U$. Consequently, the corresponding extrinsic LLR of the VN is

$$L_{ext} = ln(\frac{P_{ext}}{1 - P_{ext}}) = ln(\frac{U + V}{U - V}), \quad (4)$$

where $ln(\cdot)$ indicates the natural logarithm operation.It should be noted that the contents of the saturating counters can also be used for the HDD method used in RS decoding. In this case, a hard-decision is applied to V. This usage is similar to the previous usage of up/down counters in [5], [6], [9], [10] where the sign-bit of the counter determines the hard-decision. Thus, a '0' sign-bit indicates a -1 decision and, a '1' sign-bit indicates a $+1$ decision for a BPSK transmission.

C. The Summary of the Proposed Decoding Method

The proposed method starts with the binary PCM, H_b, and $\mathbf{L_{ch}}$. At the adaptation step, the reliability-based PCM adaptation is performed to obtain $H_b^{(l)}$ as discussed in the Section II-B. After the adaptation step, the LLRs are scaled by NDS according to (1) and then transformed to stochastic bit streams. Stochastic decoding is then applied to the adapted PCM $H_b^{(l)}$ and proceeds by VNs and PNs exchanging bits for a fixed number of DCs. At the end of the last DC, the contents of up/down counters are used as the extrinsic LLRs, $\mathbf{L}_{ext}^{(l)}$, as mentioned in the subsection III-B. The $\mathbf{L}_{ext}^{(l)}$ is then used to update LLRs according to the ABP or the TAB. For the case of RS decoding, in addition to soft-output information, the HDD is also applied to the contents of counters as mentioned in the subsection III-B. The decoding process continues unless it has been run for a fixed maximum number of iterations, l_{max}, or all the parity checks are satisfied sooner.

IV. Simulation Results for Linear Block Codes

In this section, the BER or the Frame Error Rate (FER) performance of linear block codes are presented. Concerning BCH codes, extended BCH codes are considered since they are more efficient than non-extended codes. A BPSK transmission with $Y = 6$ and an average bit energy of E_b over an AWGN channel is assumed for each simulation. The PCM adaption step for BCH and RS codes is done based on the ABP. This step for block turbo codes is performed based on the TAB.

A. BCH Codes

The BER performance of a (128,120) BCH code is depicted in Fig. 7. For this code, three decoding methods are successfully employed: the ABP, the Maximum A Posteriori (MAP) probability decoding and the stochastic decoding. For comparison, the uncoded BPSK and the Maximum-Likelihood (ML) lower bound [15] curves are also plotted. The values of the damping coefficient and number of adaptations are $\lambda = 0.2$ and $l_{max} = 5$. The EM length of $M = 25$ is used for the proposed stochastic decoding method. Stochastic decoding runs for a fixed number of 1K DCs. No significant BER deviation is observed for (128,120) BCH codes between the stochastic decoding and floating-point ABP. This result was also observed for other BCH codes (not shown). The MAP decoding outperforms these two methods by about 0.25 dB at a BER of 10^{-6} and achieves the asymptotic bound. However, the MAP decoding of this BCH code requires a trellis with 256 states and 128 sections which is too complex for implementation.

B. Reed-Solomon Codes

Figs. 8 and 9 respectively show the simulation results obtained for (31,25) and (63,55) RS codes over $\mathrm{GF}(2^5)$ and $\mathrm{GF}(2^6)$. In each figure, FER performance for Algebraic hard-decision decoding method, the ABP and, the stochastic decoding method are shown. The value of the damping coefficient and number of adaptations for the (31,25) RS code are $\lambda = 0.05$ and $l_{max} = 20$ and, for the (63,55) RS code they

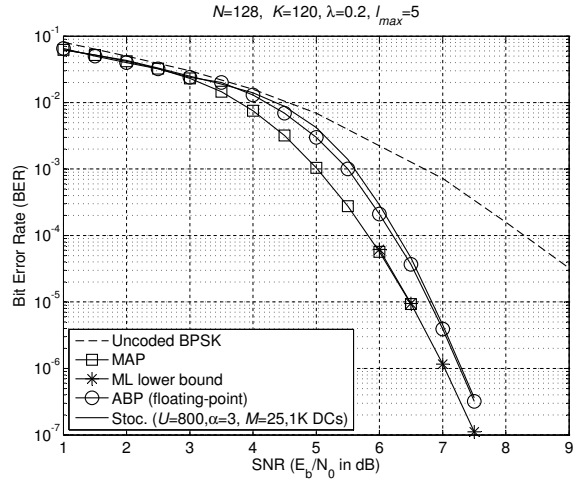

Fig. 7. Simulation results for a (128,120) BCH code.

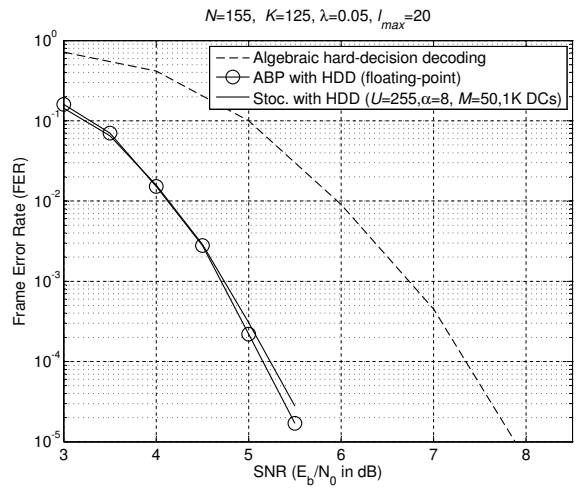

Fig. 8. Simulation results for a (31,25) RS code over $\mathrm{GF}(2^5)$.

are $\lambda = 0.12$ and $l_{max} = 5$. The NDS parameters used in stochastic decoding is $\alpha = 8$ for both codes[2]. The EM length of $M = 50$ and $M = 64$ are used for decoding the (31,25) code and the (63,55) code, respectively. Stochastic decoding runs for 1K DCs between each adaptation for the (31,25) RS code and, for the (63,55) code, it runs for 2K DCs between each PCM adaptation. Similar to [11], after each iteration, l, the HDD is applied. The performance loss for the (31,25) code is about 0.1 dB at a FER of 3×10^{-5} and for the (63,55) code, it is about 0.13 dB at a FER of 5×10^{-5}.

C. BCH Block Turbo Codes

Results for a (256,121) block turbo code based on (16,11) BCH component decoders and a (1024,676) block turbo code based on (32,26) BCH component decoders are shown in Figs. 10 and 11. For turbo decoding of these codes, the traditional CP algorithm (with 6 global iterations and 16 error patterns), the TAB and the stochastic decoding method (applied to the TAB algorithm) are successively employed for the SISO

[2]Based on the stochastic performance of the RS codes, we found that α does not necessarily need to be smaller than Y.

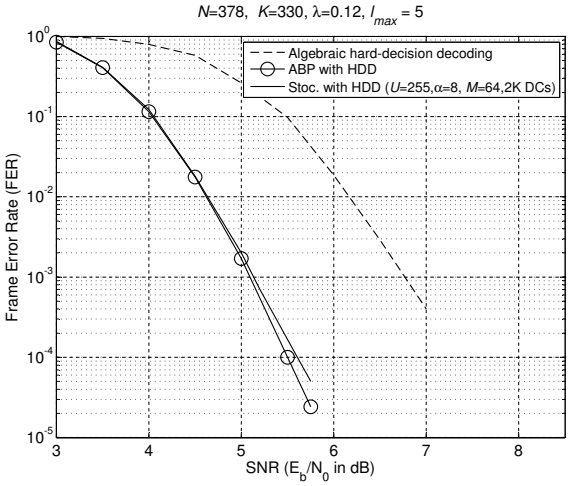

Fig. 9. Simulation results for a (63,55) RS code over GF(2^6).

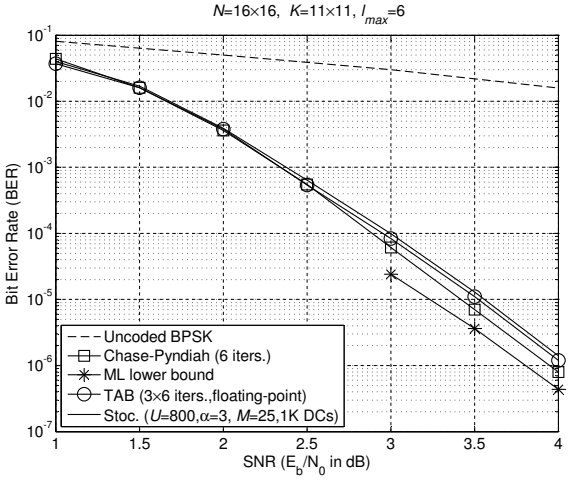

Fig. 10. Simulation results for a (256,121) BCH block turbo code.

decoding algorithm during the iterative process. Note that only 3 local iterations are necessary during the BP process of the TAB algorithm and 6 global iterations are sufficient for the two decoding methods based on TAB algorithm. In addition, no damping coefficient is necessary for the TAB algorithm. Instead, the reduction of the extrinsic information effect is done during the soft information computation [12]. The EM lengths of $M = 25$ and $M = 40$ are used for the (256,121) and the (1024,676) block turbo codes, respectively. A fixed number of 1K DCs are used for both codes. For the (256,121) turbo code, the results for the floating-point TAB and the stochastic decoding method are close and show a decoding loss of about 0.1 dB compared to the classical CP decoding at 10^{-6} BER. For the (1024,676) turbo code, the decoding loss of stochastic decoding at 10^{-6} BER is about 0.1 dB and 0.3 dB, compared to the floating-point TAB and the CP decoding, respectively.

V. Conclusions

The article extends the application of the stochastic decoding to the important classes of BCH, RS and block turbo codes. Simulation results provided for the stochastic decoding of these codes show performance close to the floating-point ABP

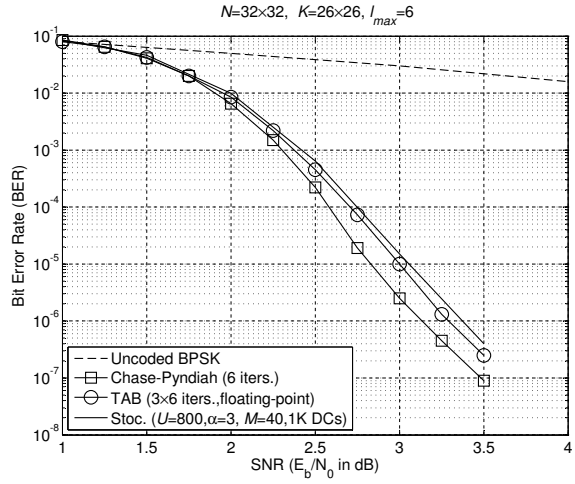

Fig. 11. Simulation results for a (1024,676) BCH block turbo code.

and/or TAB. It should be noted that stochastic decoding is also able to offer attractive complexity and performance tradeoffs. For example, it is possible to tradeoff EMs length or number of DCs with the decoding time and/or performance. Nevertheless, stochastic decoding opens up interesting research possibilities.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error correcting coding and decoding: Turbo codes," in *IEEE Int. Conf. on Communications*, May 1993, pp. 1064–1070.

[2] F. Kschischang, B. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, Feb 2001.

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

[4] V. Gaudet and A. Rapley, "Iterative decoding using stochastic computation," *Electron. Lett.*, vol. 39, no. 3, pp. 299–301, Feb. 2003.

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

[6] W. J. Gross, V. Gaudet, and A. Milner, "Stochastic implementation of LDPC decoders," in *the 39th Asilomar Conf. on Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 2005.

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

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

[9] S. Sharifi Tehrani, W. J. Gross, and S. Mannor, "Stochastic decoding of LDPC codes," *IEEE Communication Letters*, vol. 10, no. 10, pp. 716–718, Oct. 2006.

[10] S. Sharifi Tehrani, S. Mannor, and W. J. Gross, "An area-efficient FPGA-based architecture for fully-parallel stochastic LDPC decoding," to appear in *the Proc. of the IEEE Workshop on Signal Processing Systems (SiPS)*, Shanghai, China, Oct. 2007.

[11] J. Jiang and K. R. Narayanan, "Iterative soft-input soft-output decoding of Reed-Solomon codes by adapting the parity-check matrix," *IEEE Trans. on Information Theory*, vol. 52, no. 8, pp. 3746–3756, Aug. 2006.

[12] C. Jego and W. J. Gross, "Turbo decoding of product codes based on the modified adaptive belief propagation algorithm," in *Proc. of the IEEE Int. Symp. on Information Theory*, June 2007, pp. 641–645.

[13] R. Pyndiah, "Near optimum decoding of product codes: Block turbo codes," *IEEE Trans. on Communications*, pp. 1003–1010, Aug. 1998.

[14] S. Sharifi Tehrani, S. Mannor, and W. J. Gross, "Survey of stochastic computation on factor graphs," in *Proc. of the 37th IEEE International Symp. on Multiple-Valued Logic (ISMVL)*, Oslo, Norway, May 2007.

[15] F. Chiaraluce and R. Garello, "Extended hamming product codes analytical performance evaluation for low error rate applications," *IEEE Trans. on Wireless Communications*, no. 6, pp. 2353–2361, Nov. 2004.