

Soft-Decision Decoding of Block Codes: A Factor-Graph Approach

Yongmei Wei and Kambiz Homayounfar[†]

PHYBIT, INC.

AM Building 5F, 2-3-3 Higashi Gotanda, Shinagawa, Tokyo 141-0022, Japan

E-mail: kambiz@phybit.com

[†]PHYBIT, INC.

10-10 Cendex Center, 120 Lower Delta Road, Singapore 169208

E-mail: bijan@phybit.com

Abstract Linear block codes capable of correcting a burst of bit errors are desirable for protection of control plane data in mobile networks because of their effectiveness and low overhead. Yet most decoding schemes for block codes use hard decisions. The recent use of low-density parity-check (LDPC) decoders based on factor-graph, sum-product algorithm (SPA) promise improved performance over hard-decision block decoding for codes with large and sparse parity-check matrices. This requirement has thus far limited the application of SPA to most well-known block codes. We present a simple modification of SPA which can handle non-sparse matrices. We illustrate the advantage of this approach in the case of Reed-Solomon (RS) codes.

Key words Linear block codes, factor-graph, adaptive parity check matrix, Augmented cycle-free sub-graph, Reed-Solomon code.

1 Introduction

Linear block codes are widely used in a variety of commercial applications, such as low-density parity-check (LDPC) in WiMAX and DVB [5], and Reed-Solomon (RS) codes in data storage systems including CDs and DVDs [2]. Recently, tremendous effort has been focused on soft-decision decoding of these codes under factor-graph framework. Based on factor-graph framework, decoding is realized using an iterative algorithm called sum-product algorithm (SPA). It has been shown [1][3] that significant performance improvement has been achieved. Unfortunately, the SPA works well only when the parity check matrix of the codes is sparse. However, there exist plenty of codes which do not satisfy this requirement. For example, the parity check matrix of RS code belongs to high density parity check matrix and it is far from sparse. For the codes having non-sparse parity check matrix, the traditional SPA cannot work because the existence of a large amount of undesirable cycles in the corresponding factor graph. As a result, there is high possibility that incorrect messages are cycled in the graph and re-enforced each other, which leads to decoding failure. Because of the potential improvement brought by soft-decision decoding, a new SPA algorithm with a simple modification is proposed in the paper to deal with codes with non-sparse parity check matrix.

To combat the large amount of undesirable cycles in the graph of the corresponding linear block codes, the proposed method distinguishes undesirable cycles from the rest and eliminates most of those in the graph. Gen-

erally, this is realized in two steps. The first step is to select a group of suspicious bits according to their absolute values and syndrome pattern. The second step is to adapt the corresponding parity check matrix to remove all the cycles involved with selected suspicious bits by using Gaussian elimination and an algorithm called pseudo-random degree-2 connection.

As a result, the proposed method achieves up to 3 dB performance improvement compared with traditional SPA decoding of RS codes, and 1.5 dB compared with its hard-decision decoding under white Gaussian noise. Under Rayleigh channel, the performance improvement is up to 4 dB compared to the traditional SPA decoding and 8.5dB compared to the hard-decision decoding. Furthermore, the performance-complexity tradeoff can be controlled through changing two configuration parameters including the number of selected suspicious bits and one flag relating to number of times on adapting the parity check matrix and/or selecting selected suspicious bits.

2 Traditional SPA algorithm

If an (N, K) RS code over $GF(2^m)$ is taken as an example, its H can be represented as

$$H = \begin{bmatrix} 1 & \alpha^1 & \dots & \alpha^{N-1} \\ 1 & \alpha^2 & \dots & \alpha^{2(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^\delta & \dots & \alpha^{\delta(N-1)} \end{bmatrix} \quad (1)$$

Table 1: The traditional SPA based iterative soft-decision decoding

Input: Parity check matrix H_b ; the maximum number of iterations L_m and LLR vector for the coded bits from the channel observation.

Step 1: Initialization-Set $\eta_{i,j}^{[0]} = 0$ for all (i, k) with $H_b(i, k) = 1$. Set $\lambda_k^{[0]} = L(k)$, Set the loop counter, $l = 1$.

Step 2: Sum-product decoding algorithm

- (a) Check node update: For each (i, k) with $H_b(i, k) = 1$: Compute

$$\eta_{i,k}^{[l]} = -2 \tanh^{-1} \left(\prod_{p \in \Phi_{i,m}} \tanh \left(-\frac{\lambda_p^{[l-1]} - \eta_{i,p}^{[l-1]}}{2} \right) \right)$$

- (b) Bit node update: For $k = 1, 2, n$: Compute

$$\lambda_k^{[l]} = L(k) + \sum_{i \in \Psi_{n,k}} \eta_{i,k}^{[l]}$$

where $\Psi_{n,k}$ refers to the collection of the row indices at which the values of H_b are nonzero for k th column, $\Phi_{i,m}$ refers to the collection of the column indices at which the values of H_b are nonzero for i th row

- (c) Make a tentative decision: Set $\hat{c}_k = 1$ if $\lambda_k^{[l]} > 0$ else set $\hat{c}_k = 0$. If $H_b \hat{c}_k = 0$, stop. Otherwise, if $l < L_m$, set $l = l + 1$, go back to Step 2 for another iteration.

where is the primitive element of $GF(2^m)$ and $\delta = N - K$. Since each symbol in $GF(2^m)$ can be represented in m bits. The parity check matrix H can be mapped to a binary parity check matrix H_b with size of $(n - k)n$, where $n = Nm$ and $k = Km$.

Once the parity check matrix of RS code is created, the well-known sum-product algorithm (SPA)[4] shown in Table 1 for iterative soft-decision decoding can be employed to decode RS codes. Fig. 1 shows the performances in terms of word error rate (WER) comparison between the hard-decision algorithm and traditional SPA. It can be seen that, without improving, traditional SPA even yields worse performance compared with the hard-decision algorithm. The traditional SPA fails mainly due to the existence of cycles in the graph determined by the parity check matrix H_b . In the next Section, a short analysis is provided to distinguish the most devastating cycles from the rest.

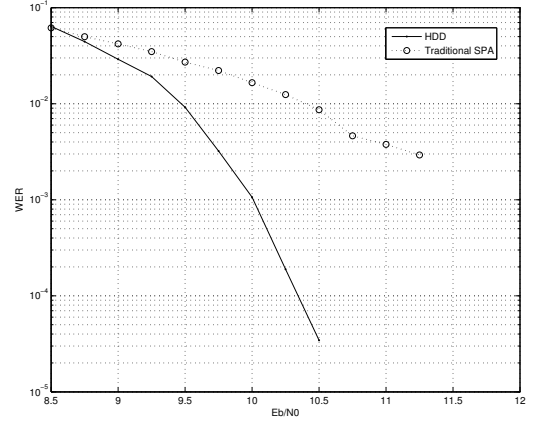


Figure 1: WER Performance comparison between hard-decision algorithm and traditional SPA for (204, 188) RS codeword.

3 Cycles Involving with More Than One Incorrect Bit

The sum-product algorithm fails mainly due to the existence of short loops determined by the parity check matrix H_b . The problem is more severe when more than one incorrect bit is involved in a loop. For example, if there are two incorrect bits within one loop and all other bits involved with the same checks are correct, the sign of η_{ik} in Table 1 are always wrong and the LLR moves towards the wrong direction in each iteration. In this case, both of the two incorrect bits enforce each other and themselves in the wrong direction, which leads to decoding failure. Thus, it is important to remove the cycles of the factor-graph involving more than one incorrect bit determined by the parity check matrix.

For the parity check matrix H_b of RS code, it can be easily observed that it contained a huge number of cycles due to its high density. For example, the row and column weights for (204, 188) RS code with $n - k = 128$ is around 810 and 60, respectively. It is difficult maybe impossible to remove all the cycles. Since it has already been analyzed that the loops involving incorrect bits are more damaging, especially for those involving with more than one incorrect bits, it is expected that all these damaging loops should be removed in order to achieve good performance. In the next Section, a new adaptive parity check matrix based decoding algorithm is proposed with augmented cycle-free sub-graphs. By using the proposed method to adapt the parity check matrix, almost all the cycles involving with more than one incorrect bits are removed and significant performance improvement has been achieved as a result.

It is worth mentioning that authors in [3] proposed a method to remove cycles involved with bits having low absolute values of LLR. Because, generally speaking, bits with lower absolute values of LLR have higher

probability to be incorrect bits. The method in [3] removes a portion of damaging loops described in the previous paragraph. When there are incorrect bits with high absolute values of LLR, however, the method without removing any loops involved with these incorrect bits, even creates loops in between the unreliable bits and these incorrect bits. In this case, the algorithm in [3] fails to converge to the correct codeword and its performance in terms of WER suffers from an error floor and hard-decision algorithm has to be included during each iteration of SPA.

4 The selection of suspicious bits and adaptation of the parity check matrix

The proposed method to adapt the parity check matrix in each iteration is majorly divided into two steps. The first is to locate a set of suspicious hard-decision error bits (SHDD-EB) whose reliabilities yields hard-decision errors in high probabilities. The second is to create a new parity check matrix remove all the cycles involving within all SHDD-EB during each iteration.

4.1 Determine the SHDD-EB

After reordering all the bits with the ascending order of LLR, the bits are divided into two groups. The first group is the $(n-k)$ bits which have relative low absolute values of LLR. The second group is the remaining k bits which have relative high absolute values of LLR.

It can be easily understood that bits with lower reliabilities are more prone to yield hard-decision errors than those with higher reliabilities. Thus, the SHDD-EB firstly includes $(n-k)$ bits which have lower reliabilities. Furthermore, it has been found that there are also incorrect bits with high reliabilities. Fig. 2 facilitates a better understanding of the existence of incorrect bits with high absolute values of LLR. It illustrates the percentage among 5320 RS codewords containing q incorrect bits not belonging to the $(n-k)$ most reliable bits at different E_b/N_0 . Generally, the percentage increases with the decreasing of E_b/N_0 . More importantly, there always exists at least one incorrect bit with high absolute LLR for all E_b/N_0 . As a result, the SHDD-EB secondly includes maximum of q bits which have the highest probabilities of being the incorrect bits are selected among the second group, where q represents the number of bits selected from the second group. These q bits are selected based on maximizing the correlation between the modulus-2 summation of the corresponding columns of H_b and syndrome vector. When $q = 1$, the bit is selected [6][1] according to

$$j = \arg \max_i ((H_b \cdot y)' \cdot h_i) \quad (2)$$

where y is the hard-decision of the LLR vector in each iteration and h_i is the i th column of H_b . When $q = 2$,

there are possibilities that either one or two bits are selected from the second group according to

$$\{a_0 j_0, a_1 j_1\} = \arg \max_{i,k} ((H_b \cdot y)' \cdot ((a_0 h_i) \oplus (a_1 h_k))) \quad (3)$$

where a_0 and a_1 are both either 1 or 0 determining whether j_0 and j_1 are selected respectively (1 means selected).

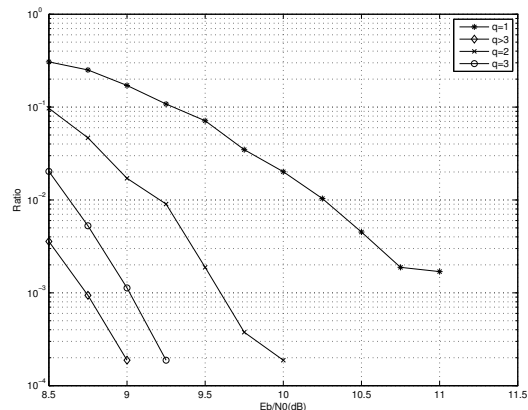


Figure 2: Ratio of (204, 188) RS codeword when there are q incorrect bits with large absolute values of LLR as a function of E_b/N_0 . Totally, 5320 RS code words are generated.

4.2 Adapt the parity check matrix

The adaption of the parity check matrix is divided into two procedures. The first step is to use Gaussian-elimination to turn the sub-matrix involving SHDD-EB with low reliabilities to a unit matrix. Through this way, the cycles involving with the $(n-k)$ bits with lower LLR, are eliminated. The second step – pseudo-random degree-2 connection, is employed to increase the column weights without creating new cycles between SHDD-EB with low LLR and SHDD-EB with high LLR. The details of pseudo-random degree-2 connection are given in Section 4.2.1.

4.2.1 Pseudo-random degree-2 connection

In this step, a pseudo-random permutation is found firstly to avoid introducing loops between the first-group SHDD-EB from the second group SHDD-EB. Then, each row of parity check matrix H_b is added to its previous row to form a new parity check matrix.

To avoid the loops between the first-group SHDD-EB and the second group SHDD-EB, it is desired that no patterns $\{1, 0, 1\}$ or $\{0, 1, 0\}$ exist in the j th column after the permutation if it is assumed that the j th column corresponds to the suspicious incorrect bits with high absolute values of LLR. An example is shown in Fig. 3 for a clearer understanding. If there are patterns like $\{1, 0, 1\}$ existed in the j th column, the parity check

matrix after adding p_{i+1} th row to p_i th row, has loops between the suspicious incorrect bits. To achieve the

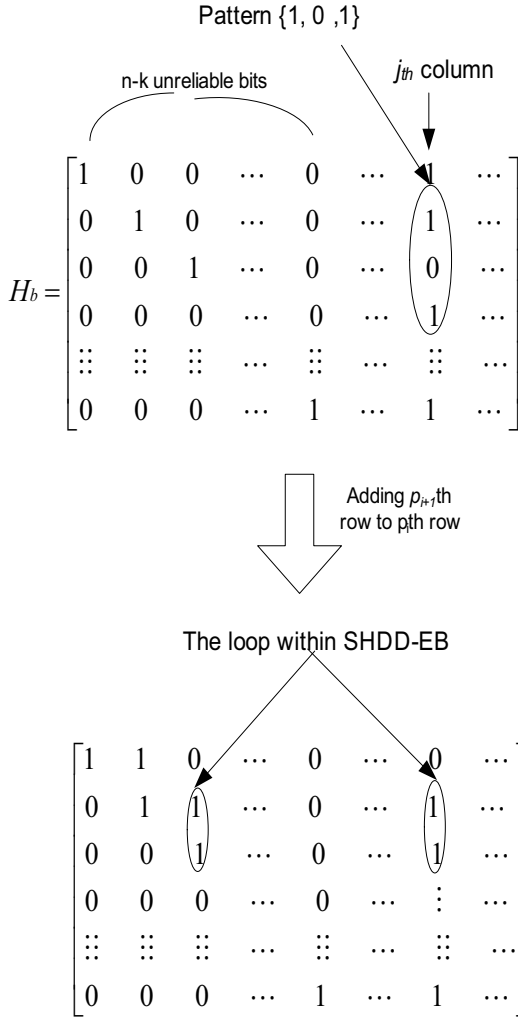


Figure 3: Loops introduced after degree-2 connection

desired permutation, the most straightforward way is to try every possible permutation, discard those which do not satisfy the above mentioned requirements until a qualified one is found. Unfortunately, this way requires an extremely heavy computational cost. Thus, a much simpler way is presented in the following for cases when $q = 1$ and $q = 2$ since it has been shown in Section 3 there are no more than two reliable incorrect bits for the most interested range of BER. It should be noticed that this method can be extended to $q = \log_2(n - k)$ in a similar way as the way that is extended from $q = 1$ to $q = 2$.

(A). When one bit in the second group of SHDD-EB is selected.

If it is assumed that j th bit is the selected bit among the second group SHDD-EB, the details of pseudo-random degree-2 connection are presented as follows,

Step 1: Order the rows of the parity check matrix H_b following the rules that the values of h_j is in a descending order, which defines a permutation t_1, t_2

\dots, t_{n-k} . Suppose that the first row of h_j being 0 is the t_r th row.

Step 2: Randomly select a number r_0 from 2 to $r - 1$ and another number r_1 from r to $n - k - 2$. Switch rows from t_{r_0} to t_{r_1} with rows from t_{r_0} to t_{r_1} , which defines the second permutation p_1, p_2, \dots, p_{n-k} .

Step 3: Add p_{i+1} th row to p_i th row, for $i = 1$ to $n - k - 1$.

Step 1 guarantees non-existence of patterns like $\{1, 0, 1\}$ or $\{0, 1, 0\}$. Step 2 introduces randomness and guarantee the bits among the second group SHDD-EB receive extrinsic information from two instead of one check. The randomness is helpful in the sense that the different checks are involved in providing different correction information in each iteration. This way helps to avoid that the algorithm gets stuck if there are incorrect bits associated with the check information.

(B)When two bits in the second group of SHDD-EB are selected,

If it is assumed that j_0 th and j_1 th bits are the selected bits among the second-group SHDD-EB, the steps of pseudo-random degree-2 connection are shown as follows,

Step 1: Reorder the rows of the parity check matrix H_b following the rules that the values of h_{j_0} is in a descending order, which defines a permutation t_1, t_2, \dots, t_{n-k} . Suppose that the first row of h_{j_0} being zero is the t_r th row.

Step 2: Reorder the rows t_1, t_2, \dots, t_{r-1} of the parity check matrix H_b following the rules that the values of h_{j_1} for these rows are in a descending order, which defines another permutation t'_1, t'_2, \dots, t'_r and suppose that the first row of h_{j_1} being 0 is the t'_{k_0} th row.

Step 3: Reorder rows $t_r, t_{r+1}, \dots, t_{n-k}$ of the parity check matrix H_b following the rules that the values of h_{j_1} for these rows are in an ascending order, which defines the third permutation and suppose that the first row of h_{j_1} being 1 is the k_1 th row.

Step 4: Randomly select a number r_0 from $\max(2, k_0)$ to $r - 1$ and another number r_1 from r to $n - k - \max(2, n - k - k_0)$. Switch rows from r_0 to $r - 1$ with rows from r_0 to r_1 , which defined the last permutation p_1, p_2, \dots, p_{n-k} .

Step 5: Add p_{i+1} th row to p_i th row, for $I = 1$ to $n - k - 1$.

For the case (B), the step 1 and step 5 are exactly the same as step 1 and 3 for case 1). Step 1 guarantees the non-existence of patterns $\{1, 0, 1\}$ or $\{0, 1, 0\}$ in the j_0 th column and the Step 2, 3 guarantee that there are no such patterns introduced in the j_1 th row without introducing the patterns in the j_0 th column. The step 4 in this case has the same functionality as that of the step 2 in the case 1).

5 Complete description of the proposed algorithms

Table 2 shows the detailed steps of the proposed algorithms. For the proposed method, two configuration parameters are added to control the tradeoff between the performance and computational complexity. The first parameter is q which is the number of bits in the second group of SHDD-EB. The second one is $flag1$ and it determines whether Step 2 and 3 in Table 2 are done in each iteration or only once in the first iteration. In the next Section, detailed performance comparisons between different configurations are shown in terms of WER curves.

6 Simulation Results

Word error rates (WER) are drawn as a function of E_b/N_o . For each E_b/N_o , either 100 word errors are recorded or a total number of 1,000,160 information words are reached. The decoded word is selected according to minimizing the summation of log likelihood ratios of the different decoded bits during each iteration compared with the hard-decision of the received vector. Fig. 4 depicts the comparison between the method in [3], the methods proposed in this paper and hard decision Berlekamp-Massey (BM) algorithm [2] in AWGN channel. Proposed method 2 and 3 represent the algorithms shown in Section 5. Method 2 refers to the case when $flag1 = 1$ and Method 3 refers to the case when $flag1 = 2$ and the proposed method 1 represents the same algorithm with $flag1=0$. It can be seen that, compared with the algorithm in [3] with BM algorithm, roughly around 0.25 dB performance gain is achieved through using the proposed method when q is one, and 0.75 dB is achieved when q is two. Furthermore, the proposed methods achieved up to 1 dB performance gain compared with the method [3] without hard decision algorithm in the iteration. Besides, the performance differences are actually neglected between method 2 and 3 when q is one. This means that similar performances are achieved with much less computational complexity. Thus, only the performances of methods 1 and 3 are shown when q is two. For the Rayleigh channel, we observe similar results. Only proposed methods 3 and 1 are shown because method 2 does not have the good tradeoff property compared with method 1 and 3. It can be seen from Fig. 5 that around 8.5 dB performance improvement is achieved compared with the BM algorithm, 3.5dB compared with the traditional SPA without adapting the parity check matrix, up to 1dB performance improvement is achieved by the proposed method 2 with $q = 1$ and $q = 2$ compared with the algorithm in [3] with BM algorithm, and 1.5 dB compared with the algorithm in [3] with BM algorithm.

Table 2: The adaptive parity check matrix based decoding with augmented cycle-free sub-graph.

Input: Parity check matrix H_b ; the maximum number of iterations L_m and LLR vector for the coded bits from the channel observation; $flag1$.

Step 1: Initialization-Set $\eta_{i,j}^{[0]} = 0$ for all (i, k) with $H_b(i, k) = 1$. Set $\lambda_k^{[0]} = L(k)$, Set the loop counter, $l = 1$.

Step 2: Parity check matrix adaptation:

- (a) Order the coded bits according to the absolute values of $\{\lambda_k^{[l]}\}$
- (b) Implement Gaussian elimination to systematize the $(n - k)$ columns of H_b which is associated with the $(n - k)$ unreliable bits.

Step 3: Parity check matrix adaptation through pseudo-random degree-2 connection algorithm

- (a) Finding SHDD-SB according to Eq. 2 or 2).
- (b) Pseudo random connection algorithm in Section 4.2.1.

Step 4: Sum-product decoding algorithm

- (a) Check node update: For each (i, k) with $H_b(i, k) = 1$: Compute

$$\eta_{i,k}^{[l]} = -2\tanh^{-1}\left(\prod_{p \in \Phi_{i,m}} \tanh\left(-\frac{\lambda_p^{[l-1]}}{2}\right)\right)$$

- (b) Bit node update: For $k = 1, 2, n$: Compute

$$\lambda_k^{[l]} = L(k) + \sum_{i \in \Psi_{n,k}} \eta_{i,k}^{[l]}$$

where $\Psi_{n,k}$ refers to the collection of the row indices at which the values of H_b are nonzero for k th column, $\Phi_{i,m}$ refers to the collection of the column indices at which the values of H_b are nonzero for i th row

- (c) Make a tentative decision: Set $\hat{c}_k = 1$ if $\lambda_k^{[l]} > 0$ else set $\hat{c}_k = 0$. If $H_b \hat{c}_k = 0$, stop. Otherwise, if $l < L_m$, set $l=l+1$. If $flag1 == 0$ go back to Step 2 for another iteration, if $flag1 == 1$ go back to Step 3 for another iteration, $flag1 == 2$ go back to Step 4 for another iteration.
-

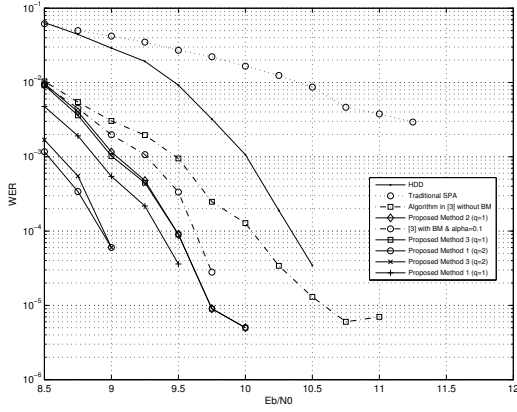


Figure 4: WER of (204, 188) RS code as a function of E_b/N_o in AWGN channel.

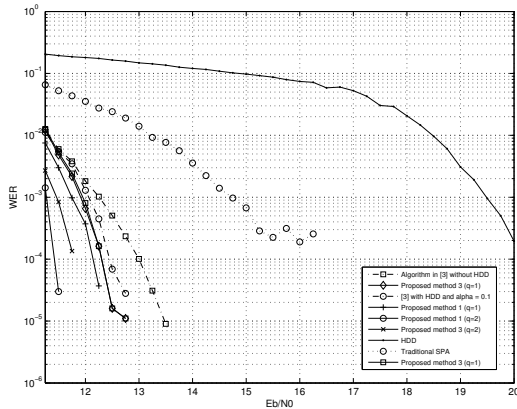


Figure 5: WER of (204, 188) RS codes as a function of E_b/N_o in Rayleigh channel.

7 Conclusion

A new iterative soft-decision decoding algorithm is proposed for decoding linear block codes. It is based on adapting the parity check matrix in each iteration. Compared with the existing adaptive parity check matrix based algorithm, the proposed one removes loops not only between the bits with low absolute values of LLR, but also between the suspicious incorrect bits with high absolute values of LLR. As a result, the performance in terms of WER is improved and the error floor at high E_b/N_0 is removed.

References

[1] A. Ahmed and N. R. Shanbhag R. Koetter. Performance analysis of the adaptive parity check matrix based soft-decision decoding algorithm. *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, 2:1995–1999, 2004.

[2] R. E. Blahut. *Algebraic codes for data transmission*. Cambridge, UK, 2002.

[3] Jing Jiang and Krishna R. Narayanan. Iterative soft-input-soft-output decoding of reed-solomon codes by adapting the parity check matrix. *IEEE Trans Information Theory*, 52(8):3746–3756, 2006.

[4] T. K. Moon. *Error correction coding: Mathematical methods and algorithms*. Wiley-Interscience, 2005.

[5] T. Ohtsuki. Ldpc codes in communications and broadcasting. *IEICE Transactions on Communications*, E90-B(3):440–453, 2007.

[6] N. Varnica, M. P. C. Fossorier, and A. Kavcic. Augmented belief propagation decoding of low-density parity check codes. *IEEE Trans on Communications*, 55(7):1308–1317, 2007.