Trellis based Extended Min-Sum for Decoding Nonbinary LDPC codes

Erbao Li*, Kiran Gunnam† and David Declercq*
*ETIS ENSEA/University of Cergy-Pontoise/CNRS UMR-8051,
6, aveune du Ponceau, F-95000, Cergy-pontoise, France.E-mail: erbao.li,Declercq@ensea.fr
† Texas A&M University, E-mail: kgunnam@ieee.org

***Abstract*—In this paper, we propose a new method to implement the Extended Min-Sum (EMS) algorithm based on trellis representation of inputting messages to the check node. The original EMS reduces the decoding complexity by choosing only n_m most reliable values and by introducing the idea of configuration sets $conf(n_m, n_c)$, where n_c is the number of deviations from the most reliable output configuration. We propose to work directly on the deviation space by building a configuration trellis based on delta messages, which serves as a new reliability measure. The algorithm is called trellis-EMS (T-EMS). By using the new trellis representation, only n_c+1 minimum values from each row of the trellis need to be considered, which reduces the ordering complexity. The trellis representation in the deviation space also reduces the number of output configurations tested, without any performance degradation. Our method is especially suitable for moderate field size q and large check node degree nonbinary LDPC codes.**

***Index Terms*—Extended Min Sum, Low Density Parity Check codes, message passing decoder, trellis representation.**

I. Introduction

Low-density parity-check (LDPC) codes were proposed by R.G Gallager [1] in 1963 and were rediscovered by D.J.C Mackay [2] in 1996. Binary LDPC codes solutions have been proposed in many standards and technologies (DVB, WiMax, Magnetic Recording storage, etc.) [3], [4]. However, binary LDPC codes have shown some weaknesses when the code length is small or moderate, or when high order modulation is used for transmission like in MIMO/OFDM systems. For these reasons, nonbinary LDPC (NB-LDPC) codes designed on high order Galois field $GF(q)$ have great potential advantages.

NB-LDPC codes are however much more complex to decode. In [5], the authors used the probability domain belief propagation (BP) decoding algorithm for NB-LDPC, whose complexity is dominated by $\mathcal{O}(q^2)$. A Fast Fourier Transform (FFT) domain implementation of BP reduces the complexity to $\mathcal{O}(q\,log(q))$. However the FFT implementation is only interesting in the case of an implementation in the probability domain. As shown in [6], an implementation which uses logarithm likelihood ratio (LLR) message format is more suited to hardware implementation since it avoids all multiplications and divisions. As a consequence, the LLR domain implementation makes the algorithm more robust to quantization.

The complexity of LLR-BP is again dominated by $\mathcal{O}(q^2)$ operations. In [7], D. Declercq and M. Fossorier introduced a reduced complexity decoder which operates on the LLR domain, and whose complexity is reduced to $\mathcal{O}(n_m\,q)$, where $n_m \ll q$ is the number of most reliable values in each incoming message vector. This algorithm has received much attention in the recent literature [8], [9].

In this paper, based on the idea of configuration sets $conf(n_m, n_c)$ in [7], we propose a new method to implement the EMS by introducing a trellis representation of the messages at the input of the check nodes. A preliminary study of this new method can be found in [10]. For the trellis representation, we only select n_c+1 most reliable messages in each section of the trellis, then combine them to form the output message at check node, where n_c now represents the number of deviations from the most reliable configuration.

There are two advantages of implementing the EMS algorithm using the trellis representation. One is that we do not need to find n_m largest values from q-sized message vector, which reduces the complexity of the sorting procedure. This means that for larger field size q, our new method will be less complex than the original EMS. The other advantage is that T-EMS reduces the number of possible output configurations, especially for large d_c. For example, when $q = 8$, our method has less potential configurations than the original EMS for codes with check node degree d_c bigger than 3.

The paper is organized as follows. In Section II, we give a brief introduction of LLR based-BP and original EMS algorithm explained on trellis. The T-EMS algorithm based on trellis representation is described in Section III together with the complexity analysis of our new method. Then in Section IV, we give simulation results on several nonbinary LDPC codes. Finally, we draw the main conclusions in Section V.

II. Logarithm domain BP and EMS algorithms for NB-LDPC codes

Similarly to binary LDPC codes, nonbinary LDPC codes can be defined by a low density parity check matrix $H_{M\times N}$ (which is referred to as H hereinafter). In the case of nonbinary codes, each element $h_{i,j}$ in the matrix H is an element from the Galois field $GF(q)$. The kernel of the low density parity check matrix defines the nonbinary code C_H.

$$\mathcal{C}_H = \{\vec{c} \in GF(q)^N | \quad H\vec{c} \overset{GF(q)}{=} \vec{0}\}$$

As in the binary case, the rank of the matrix H is usually M (assuming full rank), in which case the code rate is $R = (N-M)/N$. For simplicity, we only deal with regular NB-LDPC codes with degree profile (d_v, d_c) [7], [8].

In this paper, we use the following notations for messages passing between nodes (as depicted in Fig 1). Let $\vec{V}_{p_i v}, i = 1...d_v$ be the vector message at the input of the variable node (VN) v of degree d_v, and $\vec{U}_{vp_i}$ be the output vector message of this node, each vector message has size q. The notation 'vp' means that message comes from variable node to permutation node (PN), 'pv' means message of the opposite direction. We define similar vector messages $\vec{U}_{p_i c}, i = 1...d_c$ and $\vec{V}_{c_i p}, i = 1...d_c$ for a check node (CN) of degree d_c. Without losing generality [7], we only calculate the d_c-th output vector $\vec{V}_{d_c p}$ from check node to permutation node.

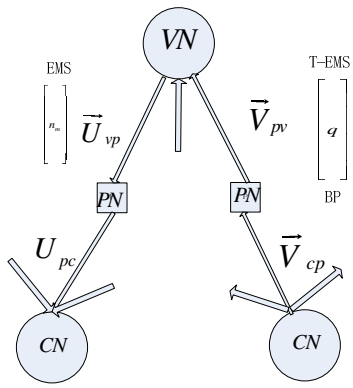

Fig. 1: message notations on the factor graph

A. LLR based-BP for NB-LDPC codes

In this section, we give a brief introduction for LLR based-BP (Log-BP). Usually, a message passing decoder for non-binary LDPC codes consists of the following five steps.

- **INITIALIZATION:** For a variable node $v(v = 1...N)$, the initial message is given as:

$$\vec{L}_v^{[0)}[x] = \ln \frac{\Pr(X_v = x/y_v)}{\Pr(X_v = 0/y_v)}, \quad x \in GF(q) \tag{1}$$

  y_v is the received reliability from the channel for the v-th variable node. X_v is the v-th variable node. During the first iteration, the variable node sends message vector $\vec{U}_{vp}^{(0)} = \vec{L}_v^{(0)}$ to permutation nodes connected to it.
- **VARIABLE NODE UPDATE:** Each variable node v has d_v incoming messages $\vec{V}_{pv}^{(l)} \in \mathbb{R}^q$. where l denotes the iteration number. Combining the channel value, the variable node v sends the extrinsic output message $\vec{U}_{vp}^{(l+1)}$ to its d_v adjacent permutation nodes.

$$\vec{U}_{vp}^{(l+1)}[x] = \vec{L}_v^{(0)}[x] + \sum\nolimits_{t=1, t\neq p}^{d_v} \vec{V}_{tv}[x] \tag{2}$$

  The output vector $\vec{U}_{vp}^{(l+1)}$ is then normalized, so that $\vec{U}_{vp}^{(l+1)}[0] = 0$ as

$$\vec{U}_{vp}^{(l+1)}[x] = \vec{U}_{vp}^{(l+1)}[x] - \vec{U}_{vp}^{(l+1)}[0], \quad x \in GF(q)$$

- **PERMUTATION:** The presence of non-zero values h_{vc} in the parity check matrix of the code induces the permutation of messages index [7]. The output message of a variable node is therefore permuted as :

$$\vec{U}_{pc}^{(l+1)}[x] = \vec{U}_{vp}^{(l+1)}[h_{vc}^{-1} x], \quad x \in GF(q) \tag{3}$$

  After the check node update, the vector messages have to be inverted by the inverse transformation:

$$\vec{V}_{pv}^{(l+1)}[x] = \vec{V}_{cp}^{(l+1)}[h_{vc} x], \quad x \in GF(q) \tag{4}$$

- **CHECK NODE UPDATE:** For each check node c, we consider d_c incoming messages $\vec{U}_{pc}^{(l)} \in \mathbb{R}^q$. The check node c sends extrinsic output message $\vec{V}_{cp}^{(l+1)}$ to its d_c adjacent permutation nodes.

$$\vec{V}_{cp}^{(l+1)} = \boxtimes_{t=1, t\neq p}^{d_c} \vec{U}_{tc}^{(l)} \tag{5}$$

  where $\boxtimes$ denotes the convolution of two message vectors in the logarithm domain. Let us take $L_1, L_2 \in \mathbb{R}^q$ as an example, for $x \in GF(q)$, the $\boxtimes$ operation is:

$$(L_1 \boxtimes L_2)[x] = \ln\Big(\sum_{x_1, x_2 \in GF(q), x = x_1 + x_2} e^{L_1[x_1] + L_2[x_2]}\Big)$$

  We can moreover simplify the convolution operator by a simple max function, as in [6] [11]. The check node update reduces then to:

$$\vec{V}_{d_c p}^{(l+1)}[x] = \max_{x = \sum_{i=1}^{d_c - 1} x_i} \sum\nolimits_{t=1}^{d_c - 1} \vec{U}_{td_c}^{(l)}[x_t] \tag{6}$$

- **TENTATIVE DECISION:** For each variable node v, the estimated symbol $x_v^{(l+1)}$ is computed from $d_v + 1$ incoming vector messages of the adjacent permutation nodes and the channel value.

$$x_v^{(l+1)} = \arg\max_{x \in GF(q)} (\vec{\mu}_v^{(l+1)}[x]) \tag{7}$$

  where $\vec{\mu}_v^{(l+1)}$ is calculated as

$$\vec{\mu}_v^{(l+1)}[x] = \vec{L}_v^0[x] + \sum\nolimits_{p=1}^{d_v} \vec{V}_{pv}^{(l)}[x], \quad x \in GF(q)$$

For NB-LDPC codes decoding, as we can see, the most complicated part is the check node update and the complexity increases significantly as the field size q and the check node degree d_c increase. The EMS algorithm, described in the next section, is designed to reduce this large complexity.

B. Original EMS algorithm

Here, we recall some parameters taken from [7] to be used in our main algorithm. Only the check node update will be discussed since the other steps are the same as for the Log-BP. In the EMS, only the n_m largest values from each incoming q-sized message vector $\vec{U}_{pc}^{(l)}$ are considered. And the so-called *configuration set* is defined as:

$$\begin{aligned} conf(n_m) = \{ & \vec{\beta}_{\vec{k}} = [\beta_1^{(k_1)} ... \beta_{d_c - 1}^{(k_{d_c - 1})}]^T : \\ & \forall \vec{k} = [k_1 .. k_{d_c - 1}]^T \in \{1 ... n_m\}^{d_c - 1} \} \end{aligned} \tag{8}$$

k_i in the configuration set means the k_i-th largest value from i-th incoming message vector. So, the number of possible configurations is $n_m^{d_c - 1}$. The configuration set $conf(1)$ contains only one element which forms the largest output

configuration. We denote it as 0-order configuration. Then $conf(n_m)^{(l)}$ is the configuration set that differs from 0-order configuration in exactly l entries. And $conf(n_m, n_c)$ is the set of all configurations which differ at most n_c entries from 0-order configuration. It can be defined as:

$$conf(n_m, n_c) = \cup_{l=0}^{n_c} conf(n_m)^{(l)}$$

For the output message corresponding to symbol $x \in GF(q)$, its output configuration set is:

$$conf_x(n_m, n_c) = \{\vec{\beta}_{\vec{k}} \in conf(n_m, n_c) : x + \sum_{c=1}^{d_c-1} \beta_c^{(k_c)} = 0\}$$

As $conf_x(n_m, n_c)$ could be empty for some n_m and n_c, we need to include the subset $conf_x(q, 1)$ to ensure that each entry in the output vector message is filled. The final configuration set corresponding to output symbol x is then defined as:

$$\overline{Conf}_x(n_m, n_c) = conf_x(n_m, n_c) \cup conf_x(q, 1) \quad (9)$$

The check node update function (6) is replaced in the EMS by the following equation:

$$\vec{V}_{d_c p}^{(l+1)}[x] = \max_{\vec{\beta}_{\vec{k}} \in \overline{Conf}_x(n_m, n_c)} \sum_{t=1}^{d_c-1} \vec{U}_{td_c}^{(l)}[\beta_t^{(k_t)}] \quad (10)$$

The idea of configuration set reduces the number of possible output configurations we need to test. The largest reliability in $\vec{V}_{d_c p}^{(l+1)}$ and its corresponding index which we will use in the next section can be calculated as:

$$L_{max} = \sum_{p=1}^{d_c-1} \vec{U}_{pd_c}^{(l)}[\beta_p^{(1)}], \quad \beta_{max} = \sum_{p=1}^{d_c-1} \beta_p^{(1)} \quad (11)$$

Fig 2 shows the trellis representation of the considered messages for a check node degree $d_c = 5$ in $GF(4)$. Each column represents one input message, and each row represents one symbol on $GF(4)$. An example of LLR reliability is shown as a label on each symbol in this trellis. For this example, the number of the largest values in each column of the trellis is chosen as $n_m = 3$. Finally, the d_c-th maximum output (last column) configuration is represented by the path drawn in solid line, and the maximum output index is $\beta_{max} = 1 + \alpha$.

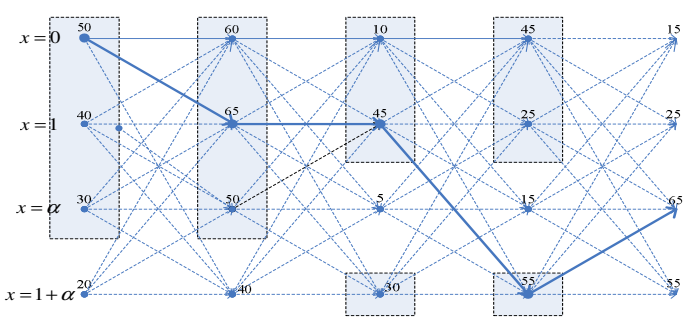

Fig. 2: Trellis representation for the original EMS with $n_m = 3$

In the original EMS, the ordering of the messages makes the algorithm computationally intensive, especially for large field size q. Indeed, the complexity of choosing the n_m largest values from a q-sized vector is $\mathcal{O}(n_m\, q)$. The T-EMS is proposed to reduce the ordering complexity and size of configuration set further, as we explain in the next section.

III. T-EMS: EMS ALGORITHM BASED ON TRELLIS

In this section we present in details the new T-EMS algorithm, which capitalizes on the trellis of Fig.2 to minimize the number of operation during the check-node update. First, we need to define the vector message domain: the *delta message domain*, which has been proposed in the literature [10], [12].

$$\Delta\vec{U}_{pc}^{(l)}[\Delta\beta = x + \beta_p^{(1)}] = \vec{U}_{pc}^{(l)}[\beta_p^{(1)}] - \vec{U}_{pc}^{(l)}[x] \quad (12)$$

In this definition, $\beta_p^{(1)}$ is the largest element's index of the vector message $\vec{U}_{pc}^{(l)}$. By substracting the maximum value to all LLRs in the vector message, we end up with only positive values in the delta messages. Fig.3 shows the new delta messages and symbol indices. With this new trellis representation, the path corresponding to the most reliable output value is always the first row of the trellis.

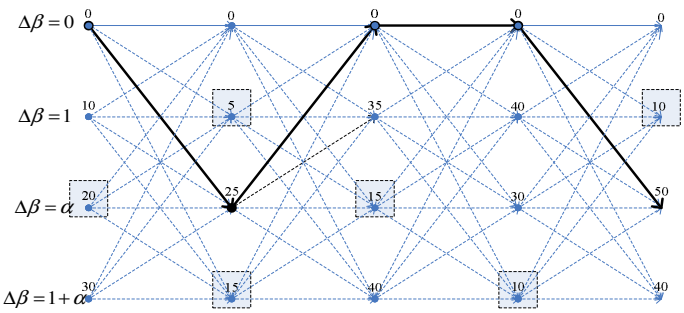

Fig. 3: Trellis on delta message domain with output $\Delta\beta = \alpha$

A. T-EMS Algorithm Description

Using equations (10), (11), (12), we can rewrite the check node output $\vec{V}_{d_c p}^{(l+1)}[x]$ as follow:

$$\begin{aligned}\vec{V}_{d_c p}^{(l+1)}[x] &= \max_{\vec{\beta}_{\vec{k}} \in \overline{Conf}_x(n_m, n_c)} \sum_{t=1}^{d_c-1} \vec{U}_{td_c}^{(l)}[\beta_t^{(k_t)}] \\ &= \max_{\vec{\beta}_{\vec{k}} \in \overline{Conf}_x(n_m, n_c)} \sum_{t=1}^{d_c-1} \Big[\vec{U}_{td_c}^{(l)}[\beta_t^{(1)}] \\ &\quad - (\vec{U}_{td_c}^{(l)}[\beta_t^{(1)}] - \vec{U}_{td_c}^{(l)}[\beta_t^{(k_t)}])\Big] \\ &= L_{max} - \min_{\vec{\beta}_{\vec{k}} \in \overline{Conf}_x(n_m, n_c)} (\sum_{t=1}^{d_c-1} \Delta\vec{U}_{td_c}^{(l)}[\Delta\beta_t^{(k_t)}])\end{aligned} \quad (13)$$

Since for each column (output vector) L_{max} is constant, we can define a new reliability variable from equation (13) that depends only on the delta messages we have defined (12). In the following of the paper, we will deal only with this new reliability measure:

$$\Delta\vec{V}_{d_c p}^{(l+1)}[\Delta\beta] = \min_{\vec{\beta}_{\vec{k}} \in \overline{Conf}_x(n_m, n_c)} (\sum_{t=1}^{d_c-1} \Delta\vec{U}_{td_c}^{(l)}[\Delta\beta_t^{(k_t)}]) \quad (14)$$

The output index $x = \sum_{t=1}^{d_c-1} \beta_t^{(k_t)}$ in equation (13) and the delta output index $\Delta\beta = \sum_{t=1}^{d_c-1} \Delta\beta_t^{(k_t)}$ in equation (14) are related by the following equation:

$$\begin{aligned}\Delta\beta &= \sum_{t=1}^{d_c-1} \beta_t^{(1)} + \sum_{t=1}^{d_c-1} \beta_t^{(k_t)} \\ &= \beta_{max} + x\end{aligned} \quad (15)$$

The solid line in Fig 3 is the case of configuration $\overline{Conf}_x(n_m, 1)$, that is the input configuration which has only one deviation from the 0-order configuration (first row in the trellis), and the corresponding output index is $\Delta\beta = \beta_t^{(k_t)}$. To form the configuration set $\overline{Conf}_x(n_m, 1)$, we select two minimum values from each row of the trellis to avoid that the minimum value could be on the output column. For $\overline{Conf}_x(n_m, 2)$, we need to choose 3 minimum messages, and so on. In Fig.4, we show the example of $\overline{Conf}_x(n_m, 2)$ for output index $x = \beta_{max} + \Delta\beta = 0$.

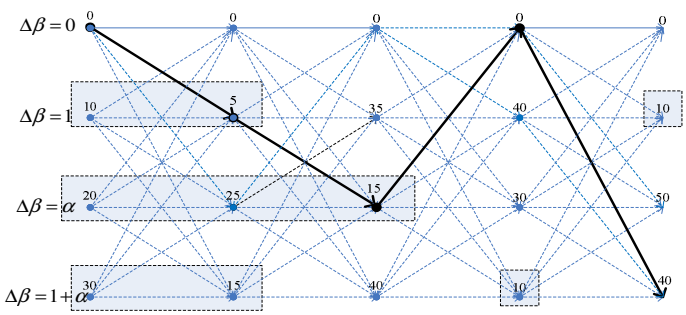

Fig. 4: Trellis for $conf(n_m, 2)$ with output $\Delta\beta = 1 + \alpha$

With the procedure described above, the number of message values we need to select from each row is $n_c + 1$ for the computation of the configuration set $\overline{Conf}_x(n_m, n_c)$. We can see that the complexity of the check-node update is no longer dependant on n_m, but only on n_c. The number of values which needs to be considered in the T-EMS is then much smaller than for the original EMS, while the final output value is exactly the same. This will be discussed in Section III-B.

During the check-node update of the T-EMS algorithm, the n_c deviations (expressed in delta values) which form an output path are computed as:

$$\Delta\vec{V}_{d_c p}^{(l+1)}[\Delta\beta] = \min\Big(\Delta\vec{V}_{d_c p}^{(l+1)}[\Delta\beta], \min_{1\le t_i\le d_c-1} \sum_{i=1}^{n_c} \Delta\vec{U}_{t_i d_c}^{(l)}[\Delta\beta_{t_i}^{(k_{t_i})}]\Big). \quad \forall i \neq j, t_i \neq t_j \tag{16}$$

$$\Delta\beta = \sum\nolimits_{i=1}^{n_c} \Delta\beta_{t_i}^{(k_{t_i})} \tag{17}$$

Finally, with equations (13) and (16), we can transform the delta message domain back to the original LLR domain by:

$$\vec{V}_{d_c p}^{(l+1)}[x] = L_{max} - \Delta\vec{V}_{d_c p}^{(l+1)}[\Delta\beta] \tag{18}$$

Note that we need to go back in the original LLR domain in order to compensate the vector message and take the tentative decision. Indeed, as in the original EMS, the check node update is followed by a compensation step, implemented by substracting the values in the vector message by the value at index zero:

$$\vec{V}_{d_c p}^{(l+1)}[x] = \Delta\vec{V}_{d_c p}^{(l+1)}[\beta_{max}] - \Delta\vec{V}_{d_c p}^{(l+1)}[\Delta\beta] \tag{19}$$

The tentative decision is then the same as in (7).

B. Complexity Analysis

The complexity reduction of our new T-EMS algorithm is twofold. The first complexity reduction relates to the ordering in the vector messages. In the original EMS, for a check node with degree d_c, we need to find the n_m largest values from the d_c q-sized vector messages, which is the bottleneck of the EMS implementation.

However, in the T-EMS — except for finding the maximum value from each input message vector which is used for delta message calculation, — the messages of size n_m are not sorted. Instead, we need to choose $n_c + 1$ minimum values from a d_c-sized vector, and this $q-1$ times. Roughly speaking, we reduce the complexity in the ordering part of the algorithm by a factor $n_m/(n_c + 1)$. Since it is advised in [7] to choose larger n_m for higher field but very small value for n_c, this means that for large field size q, our algorithm is very much less complex than the original EMS.

The other aspect in which we gain complexity is when forming the output configuration sets. In the original EMS, we consider the n_m largest values from each message vector, stored in the columns of the trellis. Then the number of total messages we deal with is $m_o = n_m \times d_c$. However, in the T-EMS, the number of messages is reduced to $m_t = (n_c + 1) \times (q - 1)$. Let us now illustrate the impact of this reduced number of considered values on the number of explored paths in the trellis for both algorithms.

For the original EMS, the total number of possible paths is [7]:

$$nb_o = \sum\nolimits_{k=0}^{n_c} C_{d_c-1}^{k} \times (n_m - 1)^k \tag{20}$$

For the T-EMS, the deviations are chosen from different rows and columns of the trellis which reduces the number of possible paths. Let us take $n_c = 2$ as an example. If we choose two deviations from the same row, the output index will be zero, and its message output could not be smaller (delta messages are all non-negative) than zero. Taking this argument into account, the total number of possible paths for T-EMS is:

$$nb_t = \sum\nolimits_{k=0}^{n_c} C_{q-1}^{k} \times C_{k+1}^{1} \tag{21}$$

In Fig.5, we show the number of possible paths for the original EMS and the T-EMS as d_c increasing. Here, we fix the parameters q and $conf(n_m, n_c)$, and as a consequence the number of possible paths for T-EMS is constant. For a small field $q = 8$, T-EMS explores fewer paths than the EMS for all d_c greater than 3. For larger field $q = 64$, our algorithm has advantage only for $d_c > 10$ (although the sorting complexity is still reduced).

Another advantage of our algorithm is that it is well adapted for hardware implementation. There are two distinct advantages: 1) Ultra-simple non-recursive check node unit implementation: The check node unit implementation with the new algorithm can now be partitioned as two non-recursive computational blocks viz., partial state processing unit and R selection unit (check node output messages generation unit). The partial state processing involves finding a partial state that comprised of the most reliable messages along the row of the trellis and then transferring the partial state to a final state register. The R selection unit now generates the check node

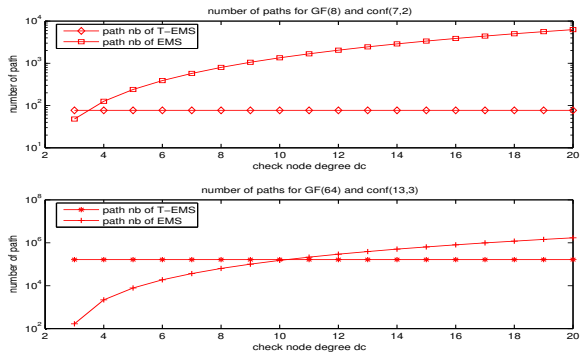

Fig. 5: number of output configurations on the trellis

unit outputs (also called R messages in other publications) from this finite final state. 2) Value-reuse property and reduction of memory requirements: Since the final state memory requirement is only $(q-1)\times(n_c+1)$ for each row in H matrix as compared to $(q-1)\times d_c$ messages. This reduces the memory required for check node messages significantly similar to binary min-sum decoders [3]. The hardware implementation issues will be discussed in more details in future papers.

IV. Simulation results

In this section, we give some simulation results on two different NB-LDPC codes, and compare T-EMS with original EMS and Log-BP algorithm. All the simulations assume transmission over a BPSK-AWGN channel. The maximum iteration number is set to 100. Fig.6 shows the performance comparison of a $GF(8)$ code with length $N_s = 155$ symbols, build from a quasi-cyclic graph with $(d_v, d_c) = (3, 5)$ [13]. The parameters of the EMS and T-EMS are $n_m = 7$ and $n_c = 2$. As we can see that T-EMS and EMS have the same performance, as expected. Fig.7 shows the result for a $N_s = 192$ NB-LDPC code on $GF(64)$. Its degree distribution is $(d_v, d_c) = (2, 8)$. The parameters of EMS and T-EMS are $n_m = 13$ and $n_c = 3$. Both simulation results are done without offset or scale compensation since the goal of this paper is mainly to show that the T-EMS has the same performance as the original EMS, but with less complexity.

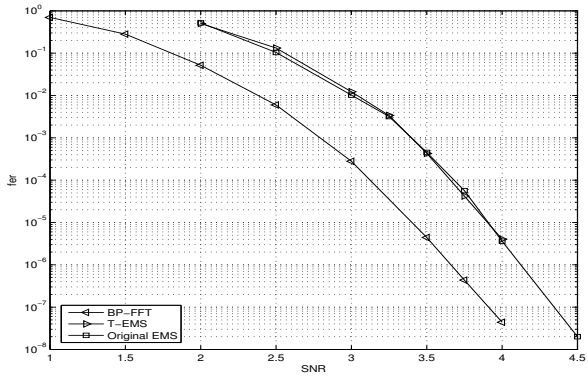

Fig. 6: performance comparison on $GF(8)$, max iter=100

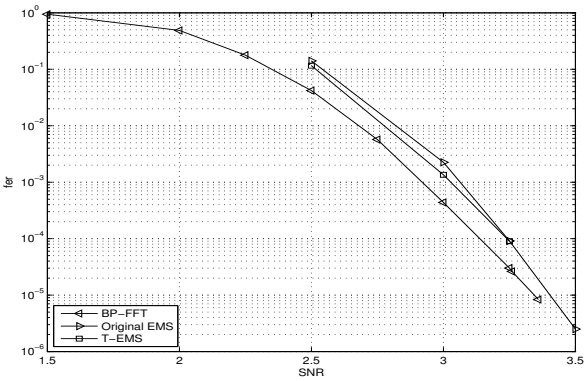

Fig. 7: performance comparison on $GF(64)$, max iter=100

V. Conclusion

In this paper, we have introduced a new trellis based implementation of the EMS using transformed delta messages. Our new algorithm reduces the computational complexity of the EMS in two aspects without losing performance. Our new implementation is especially interesting for high rate codes (large values of d_c), which is of special interest for magnetic storage recording channels and high rate wireless channels.

Our new implementation method is suitable for layered decoder structure which has less decoding delay and memory requirement. The comprehensive treatment of our new trellis based implementation on the hardware structure will be covered in follow-up papers.

References

[1] R. G. Gallager, "Low-density parity-check codes," *Cambridge, MA: MIT. Press*, 1963.

[2] D. MacKay, "Good error correcting codes based on very sparse matrices," *IEEE Trans. inform Theory*, vol. vol.46, pp. 399–431, Mar 1999.

[3] K. K. Gunnam, G. S. Choi, M. B. Yeary, and M. Atiquzzaman, "VLSI architectures for layered decoding for irregular LDPC codes of WiMax.," *In Proc. of IEEE International Conference on Communications (ICC'07)*, pp. 4542–4547, June 2007.

[4] K. K. G. et al, "Next generation iterative LDPC solutions for magnetic recording storage," *in proceedings of 42nd Asilomar Conference on Signals, Systems and Computers (ACSSC)*, pp. 1148–1152., Oct 2008.

[5] M. Davey and D. MacKay, "Low density parity check codes over $GF(q)$," *IEEE Commun. Lett*, vol. vol 2, pp. 165–167, Jun 1998.

[6] H. Wymeersch, H. Steendam, and M. Moeneclaey, "Log-domain decoding of LDPC codes over $GF(q)$," *In Proc. IEEE International Conference on Communications (ICC)*, pp. 772 – 776, Jun 2004.

[7] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over $GF(q)$," *IEEE Trans. Commun.*, vol. 55, pp. 633–643, Apr 2007.

[8] A. Voicila, D. Declercq, F. cois Verdier, M. Fossorier, and P. Urard, "Low-complexity decoding for non-binary ldpc codes in high order fields," *IEEE Transactions on Communications*, vol. 58 Issue 5, May 2010.

[9] Y. Sun, Y. Zhang, J. Hu, and Z. Zhang, "FPGA implementation of non-binary quasi-cyclic LDPC decoder based on EMS algorithm," *ICCCAS, Milpitas, CA*, July 2009.

[10] K. K. Gunnam, G. S. Choi, and M. B. Yeary, "Value-reuse properties of min-sum for $GF(q)$," *Texas A&M University Technical Note*, Oct. 2006.

[11] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X.-Y. Hu, "Reduced complexity decoding of ldpc codes,," *IEEE Trans.Commun.*, vol. vol.53, Aug 2005.

[12] V. Savin, "Min-max decoding for non binary LDPC codes," *IEEE Int. Symp. Inf. Theory, Toronto, Canada*, Jul 2008.

[13] R. M. Tanner, D. Sridhara, and T. Fuja, "A class of group-structured. ldpc codes," *in ISCTA , Ambleside, England*, 2001.