Fourteenth International Workshop on Algebraic and Combinatorial Coding Theory
September 7–13, 2014, Svetlogorsk (Kaliningrad region), Russia *pp. 336–341*

On the Decoding of Tail-Biting UM-LDPC Codes [1]

Igor Zhilin zhilin@iitp.ru
Pavel Rybin prybin@iitp.ru
Fedor Ivanov fii@iitp.ru
Victor Zyablov zyablov@iitp.ru
IITP RAS

Abstract. This paper deals with tail-biting Unit Memory codes based on Low-Density Parity-Check block codes (UM-LDPC codes). Three different decoding algorithms of this construction are considered. The first one considers the tail-biting UM-LDPC code as block code, the other two use convolution structure of this code construction. At the end of the paper simulation results for considered decoding algorithms are represented and analyzed.

1 Introduction

Unit Memory (UM) codes were introduced by Lee in 1976 [5]. These are convolutional codes with rate $R = k/n$, memory $m = 1$ and overall constraint length $\nu \leq k$. In the case when $\nu < k$ the latest codes are called Partial Unit Memory (PUM) codes. (P)UM codes are constructed based on block codes, e.g. Reed-Solomon (RS) [6], [3], BCH codes [1], or Low-Density Parity-Check (LDPC) codes [4] .The use of block codes makes an algebraic description of these convolutional codes possible and simplifies their study.

In this contribution we consider tail-biting UM codes based on LDPC block codes [2] and their decoding.

The paper is organized as follows. In section II we describe construction of considered (P)UM-LDPC codes. Then we propose several decoding algorithms for this code construction in section III. In section IV we represent simulation results for proposed decoding algorithms and analyze them.

2 Construction

Let us consider the construction of the parity-check matrix $\mathbf{H}'$ of UM-LDPC code based on set of Gallagers LDPC block codes [2]. We will denote parity-check matrices of these component codes as $\mathbf{H}_{i,0}, \mathbf{H}_{i,1},\ i = \overline{1..t}$, where t is

[1]This research has been supported by RFBR, research project No. 14-07-31197.

a period. Then UM-LDPC code is defined by its semi-infinite parity-check matrix $\mathbf{H}'$:

$$\mathbf{H}' = \begin{pmatrix} \ddots & \ddots & & & \\ & \mathbf{H}_{t,1} & \mathbf{H}_{t,0} & & \\ & & \mathbf{H}_{1,1} & \mathbf{H}_{1,0} & \\ & & & \ddots & \ddots \end{pmatrix}$$

where $\mathbf{H}_{i,0}$, $\mathbf{H}_{i,1}$ are $r \times n$ matrices, $r = n - k$. For either UM or PUM codes, block matrix $\mathbf{H}_{i,0}$ must have full rank and $\mathbf{H}_{i,1}$ may have lesser rank if the code is PUM: $rank\,(\mathbf{H}_{i,0}) = r$, $rank\,(\mathbf{H}_{i,1}) = r_1 \leq r$. The code rate R' of constructed UM-LDPC code is equal to code rate of code with parity-check matrix $\mathbf{H}_{i,0}$.

Then tail-biting UM-LDPC code with length $N = nt$ have the following parity-check matrix $\mathbf{H}$ (with tail-biting on the period t):

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_{1,1} & \mathbf{H}_{1,0} & & & \\ & \mathbf{H}_{2,1} & \mathbf{H}_{2,0} & & \\ & & \ddots & \ddots & \\ \mathbf{H}_{t,0} & & & & \mathbf{H}_{t,1} \end{pmatrix}$$

with size $tr \times tn$ (according to the condition on UM-LDPC code).

So, the code rate R of tail-biting UM-LDPC is given by:

$$R \geq 1 - \frac{tr}{tn} = 1 - \frac{r}{n} = R_{i,\{0,1\}}.$$

Remark 1. *It is important to note that constructed tail-biting UM-LDPC code is itself also LDPC code with special construction of parity-check matrix.*

Let us also introduce the following notation:

$$\mathbf{H}_i = \begin{pmatrix} \mathbf{H}_{i,1} & \mathbf{H}_{i,0} \end{pmatrix},$$

where $\mathbf{H}_i$ is parity-check matrix of LDPC code, constructed by concatenation of parity-check matrices $\mathbf{H}_{i,1}$ and $\mathbf{H}_{i,0}$.

3 Decoding algorithms

This section is devoted to the descriptions of the considered decoding algorithms of UM-LDPC codes described above. The input of these decoding algorithm are the received sequence $\mathbf{r}$ obtained after transmission of code word $\mathbf{c}$ over the binary memoryless channel with Additive White Gaussian Noise (AWGN). According to the construction of UM-LDPC codes the received sequence $\mathbf{r}$ can be represented as $\mathbf{r} = \begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \ldots & \mathbf{r}_t \end{pmatrix}$, where $\mathbf{r}_i$ is a vector with length n, that corresponds to the parity-check matrices $\mathbf{H}_{i,1}$ and $\mathbf{H}_{((i-1) \bmod t)+1,0}$ of soft-decisions of received sequence (log-likelihood ratios (LLR)).

3.1 Decoding algorithm $\mathcal{A}$

As it mentioned above the tail-biting UM-LDPC code is itself also LDPC code. So, the first decoding algorithm $\mathcal{A}$ is well-known "belief propagation" decoding algorithm of LDPC code. In this case the parity-check matrix $\mathbf{H}$ of tail-biting UM-LDPC code is considered whole as parity-check matrix of some LDPC code and is decoded using classical "belief propagation" decoding algorithm. Let

$$\mathbf{y} = D_{\mathbf{H}}^{(i_{max})}(\mathbf{r})$$

denote the decoding of received sequence $\mathbf{r}$ with algorithm $\mathcal{A}$ with i_{max} iterations $(\mathcal{A}(i_{max}))$ for parity-check matrix $\mathbf{H}$ and updated LLR values $\mathbf{y}$.

The following two algorithms consider not the whole parity-check matrix $\mathbf{H}$, but the constituent parity-check matrices $\mathbf{H}_i$, $1 \le i \le t$, on each step of decoding iteration. The main difference of these decoding algorithms is the way to exchange the decisions, made during decoding of constituent codes, between these constituent codes.

3.2 Decoding algorithm $\mathcal{B}$

The main idea of the second decoding algorithm $\mathcal{B}$ is the following. The tail-biting UM-LDPC code is decoded consequentially in the manner of convolution codes decoding. The decisions, obtained while decoding on the previous step, are used as input values for overlapped part of previous and current constituent code (for not overlapped part the values from tentative sequence are used). But due to the tail-biting structure of the code number of such steps is fixed. So, the procedure is repeated from first to the last step. Thus, in this case we need to distinguish inner iterations (performed on each step) and outer iteration (number of repetition of sequence of steps). That's why we denote this decoding algorithm as $\mathcal{B}(i_{max}, j_{max})$, where i_{max} is the number of inner iterations (the number of iterations for "belief propagation" decoder) and j_{max} is the number of outer iterations (the number of times all t constituent codes are decoded).

Let

$$\mathbf{y} = D_k^{(i_{max})}(\mathbf{x})$$

denote the decoding of tentative sequence $\mathbf{x}$ with algorithm $\mathcal{A}(i_{max})$ for parity-check matrix $\mathbf{H}_k$ and updated LLR values $\mathbf{y}$. Then the description of the decoding algorithm $\mathcal{B}(i_{max}, j_{max})$ can be written in the following way:

```
1: r_k^(0) ⟵ r_k, ∀k : 1 ≤ k ≤ t
2: r_1^(1) ⟵ r_1
3: Δ_k^(0) ⟵ 0, ∀k : 1 ≤ k ≤ t
4: for j = 1 to j_max do
5:    for k = 1 to t do
6:       k_1 ⟵ k, k_2 ⟵ (k mod t) + 1
```

$\mathbf{x}_{k_1} \longleftarrow \mathbf{r}_{k_1}^{(j)}$
$\mathbf{x}_{k_2} \longleftarrow \mathbf{r}_{k_2}^{(j-1)} - \mathbf{\Delta}_{k_2}^{(j-1)}$
$\begin{pmatrix}\mathbf{y}_{k_1} & \mathbf{y}_{k_2}\end{pmatrix} \longleftarrow D_k^{(i_{max})}\left(\begin{pmatrix}\mathbf{x}_{k_1} & \mathbf{x}_{k_2}\end{pmatrix}\right)$
$\mathbf{\Delta}_{k_2}^{(j)} \longleftarrow \mathbf{y}_{k_2} - \mathbf{x}_{k_2}$
$\mathbf{r}_{k_2}^{(j)} \longleftarrow \mathbf{r}_{k_2}^{(j-1)} + \mathbf{\Delta}_{k_2}^{(j)}$
end for
end for
return $\mathbf{r}^{(j_{max})} = \left(\mathbf{r}_1^{(j_{max})}\mathbf{r}_2^{(j_{max})}\ldots\mathbf{r}_t^{(j_{max})}\right)$

One can see that on each step k of the current outer iteration j the right part $\mathbf{r}_{k_2}^{(j)}$ (that corresponds to the parity-check matrix $\mathbf{H}_{k,0}$) of current tentative sequence $\begin{pmatrix}\mathbf{r}_{k_1}^{(j)} & \mathbf{r}_{k_1}^{(j)}\end{pmatrix}$ (that corresponds to the parity-check matrix $\mathbf{H}_k$) is updated. Then the updated right part, obtained on previous step, is used as input for the left part on the current step $k+1$. So, the right parts of constituent codes are continuously updated in direction from left to right.

3.3 Decoding algorithm $\mathcal{C}$

The third decoding algorithm $\mathcal{C}$ is the modification of the decoding algorithm $\mathcal{B}$ described above. According to the description of algorithm $\mathcal{B}$ it consequentially updates LLR values of tentative sequence in the direction from left to right. The algorithm $\mathcal{C}$ can be represented as two parallel algorithm $\mathcal{B}$ with opposite directions of LLR values updating.

According to the construction of UM-LDPC code each symbol of vector $\mathbf{r}_k$, $1 \le k \le t$, are checked by two constituent codes $H_{k,1}$ and $H_{(k \bmod t)+1,0}$. Let

$$\mathbf{y} = D_k^{(i_{max})}(\mathbf{x})$$

denote as previously the decoding of tentative sequence $\mathbf{x}$ with algorithm $\mathcal{A}(i_{max})$ for parity-check matrix $\mathbf{H}_k$ and updated LLR values $\mathbf{y}$. And let $\mathbf{\Delta}_{k,1}$ and $\mathbf{\Delta}_{k,0}$ are LLR update increments corresponding to parity-check matrices $\mathbf{H}_{k,1}$ and $\mathbf{H}_{k,0}$ respectively. Then the description of the decoding algorithm $\mathcal{C}(i_{max}, j_{max})$ is the following:

$\mathbf{r}_k^{(0)} \longleftarrow \mathbf{r}_k$, $\forall k : 1 \le k \le t$
$\mathbf{\Delta}_{k,\{0,1\}}^{(0)} \longleftarrow 0$, $\forall k : 1 \le k \le t$
for $j = 1$ **to** j_{max} **do**
for $k = 1$ **to** t **do**
$k_1 \longleftarrow k$, $k_2 \longleftarrow (k \bmod t) + 1$
$\mathbf{x}_{k_1} \longleftarrow \mathbf{r}_{k_1}^{(j-1)} - \mathbf{\Delta}_{k_1,1}^{(j-1)}$
$\mathbf{x}_{k_2} \longleftarrow \mathbf{r}_{k_2}^{(j-1)} - \mathbf{\Delta}_{k_2,0}^{(j-1)}$

$\begin{pmatrix}\mathbf{y}_{k_1} & \mathbf{y}_{k_2}\end{pmatrix} \longleftarrow D_k^{(i_{max})}\left(\begin{pmatrix}\mathbf{x}_{k_1} & \mathbf{x}_{k_2}\end{pmatrix}\right)$

$\boldsymbol{\Delta}_{k_1,1}^{(j)} \longleftarrow \mathbf{y}_{k_1} - \mathbf{x}_{k_1}$

$\boldsymbol{\Delta}_{k_2,0}^{(j)} \longleftarrow \mathbf{y}_{k_2} - \mathbf{x}_{k_2}$

end for

for $k = 1$ **to** t **do**

$\mathbf{r}_k^{(j)} \longleftarrow \mathbf{r}_k^{(j-1)} + \boldsymbol{\Delta}_{k,0}^{(j)} + \boldsymbol{\Delta}_{k,1}$

end for

end for

return $\mathbf{r}^{(j_{max})} = \left(\mathbf{r}_1^{(j_{max})}\mathbf{r}_2^{(j_{max})}\ldots\mathbf{r}_t^{(j_{max})}\right)$

So, in other words on each outer iteration j all t constituent codes with parity-check matrices $\mathbf{H}_k$, $1 \leq k \leq t$, are decoded and corresponding update increments $\boldsymbol{\Delta}_{k_1,1}^{(j)}$ and $\boldsymbol{\Delta}_{k_1,1}^{(j)}$ are calculated. Then on the next iteration the sum of tentative sequence of previous iteration and update increments of current constituent code neighbors (for the left part of constituent code the update increment, obtained for the right part of constituent from previous step, is used and vice versa) is used as input LLR for current constituent code.

4 Simulation results

Simulation results were obtained for UM-LDPC code with period $t = 4$ based on such LDPC codes (2,4) with parity-check matrices $\mathbf{H}_{i,\{0,1\}}$, that parity-check matrices $\mathbf{H}_i = \begin{pmatrix}\mathbf{H}_{i,1} & \mathbf{H}_{i,0}\end{pmatrix}$ don't have cycles with length less than 4.The code rate of constructed UM-LDPC code is equal to $R = 1 - \frac{2}{4} = 0.5$, code length $N = 2032$.

The number of iteration for decoding algorithm $\mathcal{A}\left(i_{max}\right)$ was selected to be equal to 50 ($i_{max} = 50$). According to the description of decoding algorithms $\mathcal{B}\left(i_{max}, j_{max}\right)$ and $\mathcal{C}\left(i_{max}, j_{max}\right)$ each constituent code is decoded j_{max} times with decoding algorithm $\mathcal{A}\left(i_{max}\right)$. So, the number of inner i_{max} and outer j_{max} iterations should be selected such, that $i_{max}j_{max} = 50$. After extensive simulation it was found that the best decoding performance was achieved for considered UM-LDPC code under decoding algorithms $\mathcal{B}\left(3, 17\right)$ and $\mathcal{C}\left(3, 17\right)$. This fact can be explained with the following observation. According to the construction of constituent LDPC code they don't have cycles with length 4. So, we can assume that the number of independent decoding iterations of "belief propagation" decoder is at least equal to 3. Thus the number of inner iteration should be set to 3.

In fig. 1 the simulation results for UM-LDPC code with code rate R=0.5, based on the LDPC codes (2,4), under decoding algorithms $\mathcal{A}\left(50\right)$, $\mathcal{B}\left(3, 17\right)$ and $\mathcal{C}\left(3, 17\right)$ are represented.

As you can see in fig. 1 the decoding algorithm $\mathcal{A}\left(50\right)$ has the best decoding performance. The decoding algorithm $\mathcal{B}\left(3, 17\right)$ is worse than performance of

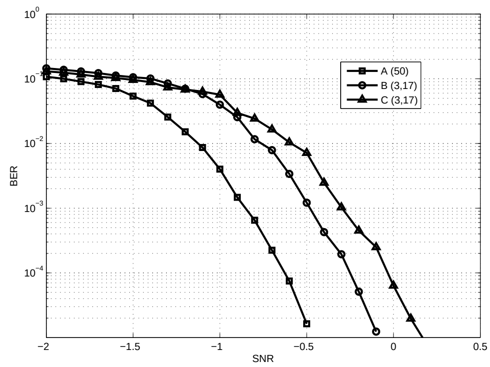

Figure 1: Simulation results for UM-LDPC code with code rate R=0.5, based on the LDPC codes (2,4), under decoding algorithms $\mathcal{A}\,(50)$, $\mathcal{B}\,(3,17)$ and $\mathcal{C}\,(3,17)$

algorithm $\mathcal{A}\,(50)$ on almost 0.4 dB at the level of bit-error rate (BER) equal to 10^{-5}. And the performance of decoding algorithm $\mathcal{C}\,(3,17)$ is worse than performance of algorithm $\mathcal{B}\,(3,17)$ on almost 0.2 dB at the level of BER equal to 10^{-5}.

Complexities of algorithms $\mathcal{B}$ and $\mathcal{C}$ is asymptotically the same as complexity of $\mathcal{A}$ up to a constant factor which is close to 1. Since $\mathcal{A}$ performs better than $\mathcal{B}$ or $\mathcal{C}$, it is unreasonable to use algorithms $\mathcal{B}$ and $\mathcal{C}$.

References

[1] U. Dettmar and U.K. Sorger. New optimal partial unit memory codes based on extended bch codes. *Electronics Letters*, 29(23):2024–2025, Nov 1993.

[2] R. G. Gallager. *Low-Density Parity-Check Codes.* MIT Press, Cambridge, MA, 1963.

[3] J. Justesen. Bounded distance decoding of unit memory codes. *Information Theory, IEEE Transactions on*, 39(5):1616–1627, Sep 1993.

[4] K. Kondrashov and V. Zyablov. On the lower bound of the free distance of partial unit memory codes based on LDPC Codes. In *Proc. IEEE Int. Symposium on Inform. Theory*, pages 1831–1835, July 2011.

[5] Lin nan Lee. Short unit-memory byte-oriented binary convolutional codes having maximal free distance (corresp.). *IEEE Trans. Inform. Theory*, 22(3):349–352, May 1976.

[6] V. Zyablov and V. Sidorenko. On periodic (partial) unit memory codes with maximum free distance. In Andrew Chmora and StephenB. Wicker, editors, *Error Control, Cryptology, and Speech Compression*, volume 829 of *Lecture Notes in Computer Science*, pages 74–79. Springer Berlin Heidelberg, 1994.