About Burst Decoding for Block-Permutation LDPC Codes

Andrei Ovchinnikov and Anna Fominykh(✉)

Saint-Petersburg State University of Aerospace Instrumentation,
B.Morskaya 67, 190000 Saint-Petersburg, Russia
mldoc@ieee.org, aawat@ya.ru

Abstract. Hard-decision decoders are considered for burst error correction for low-density parity-check codes. The decoder for block-permutation construction of low-density parity-check proposed. Experiments on complexity and error probability are conducted with burst lengths both within and beyond the burst error correction capability. Also simulation results for Gilbert model are presented.

Keywords: LDPC codes · Burst-error correcting codes · Channels with memory

1 Introduction

Nowadays, data transmission is widespread. Data is transmitted over the noisy channels that may introduce errors. To reduce the errors that appear during the transmission over the channel error correcting codes are used. One of the most promising type of error correcting codes that are widely used in modern standards are low-density parity-check (LDPC) codes [5,12] based on the block-permutation construction [15].

Errors that appear during the transmission are not independent, but most standards design codes in an assumption of errors being independent. This problem is solved during the transmission routine using the interleaving technique. One of the most important properties of LDPC codes is that they are less vulnerable to error probability decreasing when the errors tend to group in comparison, for example, to cyclic codes.

Apart from the channel transmission there are a lot of other processes, for example, processing, storing and protection during which the information may be distorted, that can be described as an artificial transmission channel that is also not free of errors grouping.

The paper was prepared with the financial support of the Ministry of Science and Higher Education and of the Russian Federation, grant agreement No. FSRF-2020-0004, "Scientific basis for architectures and communication systems development of the onboard information and computer systems new generation in aviation, space systems and unmanned vehicles".

O. Galinina et al. (Eds.): NEW2AN 2020/ruSMART 2020, LNCS 12525, pp. 393–401, 2020.
https://doi.org/10.1007/978-3-030-65726-0_35

The paper is organized as follows. Section 2 describes the simplest channel models with memory. Section 3 gives important notations about LDPC codes. Section 4 introduces decoding of error bursts for block-permutation LDPC codes. Section 5 presents the simulation results and Sect. 6 concludes the paper.

2 Channel Models with Memory

2.1 Data Transmission

When the data is transmitted over the real communication channels the influence on a signal may be described by a probability model, where the distortions of separated symbols are not independent. The simplest model that may describe such a channel is a finite state channel (Fig. 1) [6]. Through this base model several derivations may be described, including Markov model [13], Gilbert model [7] and Gilbert–Elliott model.

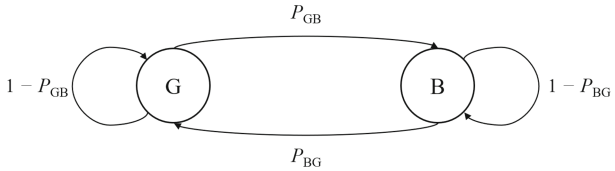

Fig. 1. Finite state channel model.

The finite state channel model describes each state as a binary symmetric channel each one with its own crossover probability that may differ. In state G (good) this probability is very low and in state B (bad) may be different depending on the channel model. In general, channel influence on the transmitted data may be described as an additive model with a received vector pointed as $\mathbf{b} = \mathbf{a} + \mathbf{e}$, where $\mathbf{a}$ is a transmitted vector, $\mathbf{e}$ is an error vector. Channel models that are under consideration describe the error vector that contains error bursts which are the areas in error vector that begin and end with one.

2.2 Information Storing

Although data transmission is a typical scenario of errors appearing, other processes are also subject to errors, moreover, such errors are not independent, and this problem can be solved by an error correcting coding. For example, distributed information storage systems allow to store data in multiple spatial distributed devices that may be organized in many different ways and may contain failures and hidden errors [11]. When such failures and errors appear, due to the system architecture the errors are more likely to form fixed length error bursts (which lengths depend on the architectural properties) rather than to spread uniformly.

2.3 Information Security

Post-quantum public-key cryptosystems become more and more relevant, most of them are based on error-correcting codes. In [10] cryptosystem based on error correcting codes that are able to correct error bursts is observed. The paper describes the encryption process that contains the generation of fixed length error burst that the legal user must be able to correct afterwards. Since the error vector is constructed in an artificial manner, the task of information protection also uses fixed length error bursts.

3 Low-Density Parity-Check Codes

Low-density parity-check (LDPC) codes were proposed by R. G. Gallager in 1962 [5], but due to the computing limitations in implementing the encoder and decoder for such codes, LDPC codes were ignored for almost 30 years. LDPC codes were rediscovered by D. Mackay [12] in the 1990s and subsequent development of computing power arose a new wave of interest to LDPC codes [14,16], which deserve attention due to the near-Shannon-limit error-correcting capability and effective procedures of encoding and decoding. These properties of LDPC codes made them desirable candidates for many communication standards [1–3]. LDPC codes are specified by a parity-check matrix that due to its sparseness may be graphically represented by means of a bipartite graph (Tanner graph). The length of the shortest cycle in the Tanner graph is called the girth. Since cycles, especially short cycles, degrade the performance of LDPC decoders, codes with large girth should be constructed.

The most popular construction of LDPC codes that allows compact representation and flexible way of code construction is a block-permutation construction [8]. A special case of block-permutation construction is Gilbert codes and their modifications which burst error correcting capability was analyzed in [9,11]. A Gilbert code is defined by a parity-check matrix $\mathbf{H}_l$,

$$\mathbf{H}_l = \begin{bmatrix} \mathbf{I}_m & \mathbf{I}_m & \mathbf{I}_m & \dots & \mathbf{I}_m \\ \mathbf{I}_m & \mathbf{C} & \mathbf{C}^2 & \dots & \mathbf{C}^{l-1} \end{bmatrix}, \tag{1}$$

where $\mathbf{I}_m$ is $(m \times m)$-identity matrix, $\mathbf{C}$ is $(m \times m)$-matrix of cyclic permutation:

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & 1 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}, \tag{2}$$

and $\ell \leq m$. Ensembles of block-permutation codes with small amount of blocks $(3 \times 6,\ 4 \times 8)$ were analyzed in [4] for information security tasks. The analysis showed that for block-permutation constructions with a block size of m maximum correctable length of the error burst is $b \leqslant m - 1$ and in the case of

codes that have no cycles of length 4 the maximum correctable burst length is distributed in the range of $[m/2, m-1]$ and is more likely to become closer to $m-1$. Moreover, it was shown that the determination of burst error correction capability has polynomial complexity and its computation may be done in an acceptable time for the codes of length up to several thousands of bits.

4 Decoding of Error Bursts for Block-Permutation LDPC Codes

In general, LDPC codes decoding methods are iterative procedures that operate on each symbol separately. Algorithms are also described as message passing between the nodes along the edges of the Tanner graph that proceeds until the predefined amount of iteration is reached or the codeword is found. For LDPC codes both hard decision and soft decision decoding algorithms may be used. The most common among them are bit flipping (BF) algorithm and belief propagation (BP) algorithm, respectively. These algorithms are able to provide low error probabilities, but do not guarantee error correction and burst error correction within the code error correction capability, that is an unacceptable scenario for many applications (for instance, data storage and data protection). Since the models that are observed in this article are discrete, only hard decision decoders will be further in consideration. It was shown in [17] that BF can't provide an acceptable error probability level in case of burst error correction, but shows better results if the windowed BF is applied instead of the original BF.

The original bit flipping algorithm for the received vector $\mathbf{y}$ and parity-check matrix $\mathbf{H}$ proceeds as follows [15]:

1. Compute syndrome $\mathbf{s} = \mathbf{y}\mathbf{H}^T$ (over $\mathbb{F}_2$), if $\mathbf{s} = \mathbf{0}$, stop, since $\mathbf{y}$ is a codeword.
2. Compute vector $\mathbf{f} = \mathbf{s}\mathbf{H}$ (over $\mathbb{Z}$).
3. Identify values of $\mathbf{f}$ that are greater than some predefined threshold (that may be chosen depending on the channel conditions), flip the corresponding bits in $\mathbf{y}$.
4. If the maximum amount of iteration is not reached, go to step 1.

The windowed version of bit flipping algorithm tries to correct burst locally if it is covered by the window. The decoder in a consequent manner considers all possible locations of burst and tries to correct errors in the currently observed window. Then, at each decoding iteration, step 3 of the original bit flipping algorithm is performed for the positions in the current window of length b. However the optimal size of the window for different channel models is an open question. It should be mentioned that when the block-permutation matrix is used BF algorithm may be modified taking into account the matrix structure. This modification includes a preprocessing step where all of nonzero syndrome elements that may be uniquely matched to the columns of $\mathbf{H}$ are eliminated. More precisely, the preprocessing step performs the following procedure: for currently observed window the syndrome elements are viewed from the first to the last.

If the currently observed syndrome element s_j is nonzero and the corresponding row in **H** on the window positions contains exactly one nonzero element, then the corresponding syndrome position is eliminated using the column of **H** that has the position of that single nonzero element in a row. After that the same procedure starts again from the first syndrome element. The preprocessing step ends, when all of syndrome positions are observed.

After such preprocessing step a situation that is shown in Fig. 2 may appear: for a given nonzero syndrome entry there is more than one nonzero element in a row, so there is more than one possible option for a specific nonzero syndrome value. Then it was proposed in [17] to switch after the preprocessing to BF within the same window. Conducted experiments showed that for block-permutation matrices all bursts with length within the code error correction capability are corrected.

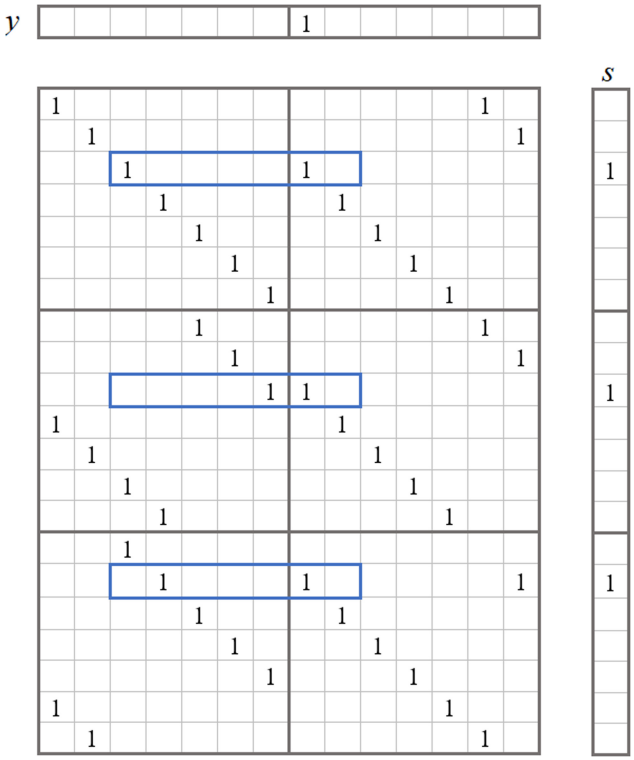

Fig. 2. After preprocessing case.

The preprocessing procedure may be followed not by BF, but by the selection from the possible variants of elements in a row. The selection procedure is described as follows: first nonzero element in the row matching to the first nonzero position in syndrome is chosen and the corresponding column is added

to the syndrome, then the first nonzero element of the row that is matched to the next nonzero syndrome position is taken and the first one is taken from that row and its column is added to the syndrome. The same is done for the rest of the nonzero positions of the syndrome. The algorithm ends when all-zero syndrome vector is reached. During the process some choices may be made wrong, so the selection process has to follow some steps back and modify the choice of some of the previously observed rows. That describes the tree like procedure that leads to time increasing, to overcome which the algorithm should be optimized. The optimization may be done as follows: selection of an element in a row should be determined by the most coincidence of one of the columns that correspond to nonzero positions in a row and syndrome. Considering such an optimization the selection process observes all the positions of the syndrome in a more efficient way. Suppose syndrome entry s_j is nonzero and the row contains more than one nonzero element. The choice of an element in a row is dictated by the most intersections of the corresponding column with a syndrome, in other words, that row element is chosen which corresponding column sum modulo two with a syndrome makes the recomputed syndrome closer to all zero vector. If several elements in a row lead to the same syndrome weight, then for each of them the recursive procedure begins that repeats previously described selection step switching to the next nonzero position in syndrome.

5 Simulation Results

The point of experiments is to determine if the observed algorithms are able to correct bursts with a length over burst error correction capability. Experiments are performed for the random block-permutation matrix with no cycles of length 4 that consists of 4×8 blocks with the block length of $m = 59$, which burst error capability $b = m - 1$ has been determined following the algorithm presented in [9]. During the experiments to perform the comparison the burst length is set to the values $b = m - 1$, $b = m + 2$ and $b = 2m + 2$. The experimental results contain time per iteration and frame error rate (FER) comparison of three algorithms. The compared algorithms are: BPBF (block-permutation bit-flip) is an algorithm with preprocessing and bit flipping, BPT (block-permutation tree) contains preprocessing and one selection, WBF (windowed bit-flip) with 1, 5 and 10 iterations. The burst of length b for an error vector is constructed in an artificial way: at first, the random start position is generated, then ones are put on the first and the last indices and the probability of one inside is p_1. The horizontal axis represents the probability of one inside burst.

It may be seen that for all burst lengths b iteration time of BPT is greater than the BPBF. BPT and BPBF are correcting every possible variant of bursts whilst $b = m-1$ and $b = m+2$, that is depicted in Fig. 3 and Fig. 5, but erroneous frames appear with $b = 2m + 2$ (Fig. 7) and FER curves of both algorithms are nearly equal. WBF frame error probability is the worst for all considered b, but it requires the smallest time to complete the iteration.

Also the experiments for the Gilbert channel model were conducted. In such experiments generating of the burst is following the next procedure: a probability

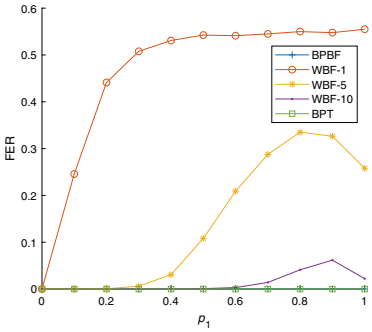

Fig. 3. FER for $b \leqslant m - 1$.

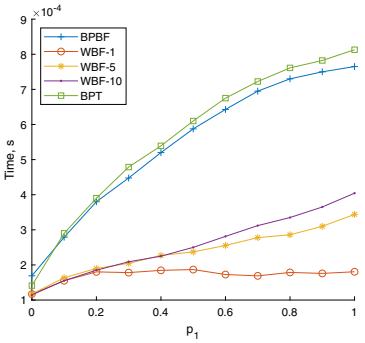

Fig. 4. Time for $b \leqslant m - 1$.

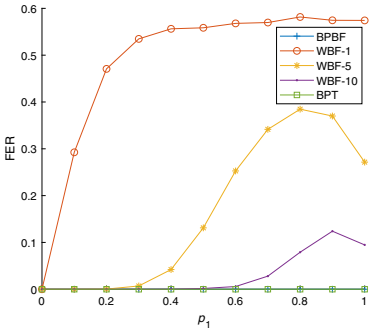

Fig. 5. FER for $b = m + 2$.

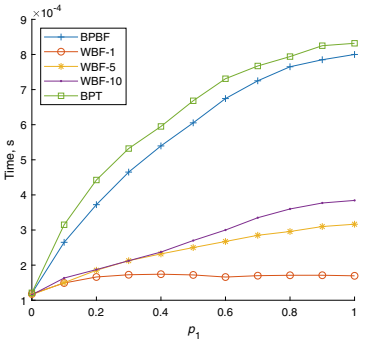

Fig. 6. Time for $b = m + 2$.

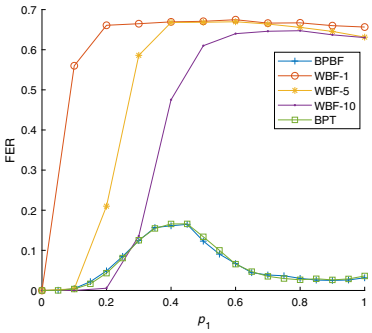

Fig. 7. FER for $b = 2m + 2$.

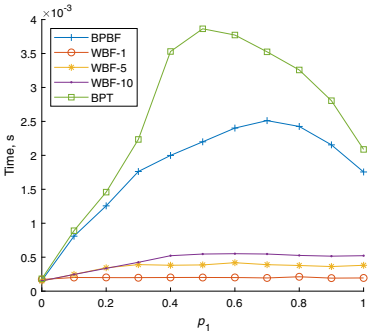

Fig. 8. Time for $b = 2m + 2$.

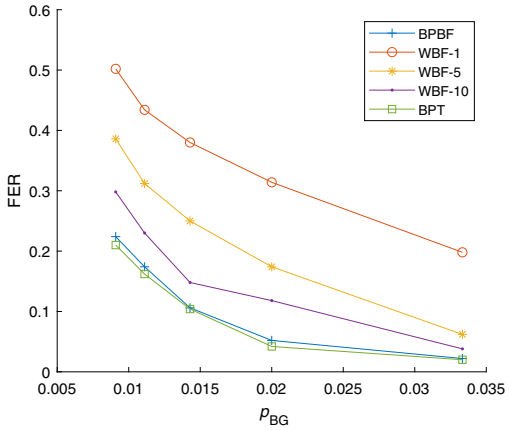

Fig. 9. Algorithm comparison for Gilbert model.

p_{BG} is determined so the mean of the length of the burst was less or equal to b (b values were chosen from the interval $[m/2; 2m + 2]$). Then random start burst position is generated, and each one inside the burst appears with probability that is set to 0.5. The modeling results are shown in Fig. 9. It may be concluded that the BPBF and BPT algorithms tend to behave similarly under the Gilbert model conditions and have a smaller error probability than WBF (Figs. 4, 6 and 8).

6 Conclusion

Hard-decision decoders were considered for burst error correction for block-permutation matrices. The modification of decoder based on optimized selection procedure is proposed. The hypothesis about the guaranteed burst error correction using the decoder with block-permutation matrices analysis is not disproved. Experiments on comparing the frame error probability were conducted with burst error correction length beyond the error correction capability. The experiments showed that for $b = m - 1$ and $b = m + 2$ the proposed algorithm (BPT) corrects all possible bursts with no much loss in time, whereas for $b = 2m+2$ BPT serves no worse than the BPBF with a small iteration time gap. Also the modeling for Gilbert model were conducted that showed the tendency of BPBF and BPT to behave similarly.

References

1. IEEE 802.11n/d1.0 Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. http://www.techstreet.cpm/standards/ieee-802-11n-2005. Accessed 10 June 2020
2. IEEE 802.16e-2005. Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems. http://www.techstreet.cpm/standards/ieee-802-16e-2005. Accessed 10 June 2020

3. Multiplexing and channel coding, 3GPP Technical Specification 38.212 V15.2.0. https://panel.castle.cloud/view_spec/38212-f11. Accessed 10 June 2020
4. Bakay, K.A., Ovchinnikov, A.A., Ilina, D.V.: About burst-correction capability of block-permutation LDPC codes ensembles. In: 2019 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF), pp. 1–4 (2019). https://doi.org/10.1109/WECONF.2019.8840134
5. Gallager, R.: Low-density parity-check codes. IRE Trans. Inf. Theory **8**(1), 21–28 (1962). https://doi.org/10.1109/TIT.1962.1057683
6. Gallager, R.: Information Theory and Reliable Communication. Wiley, New York (1968)
7. Gilbert, E.N.: Capacity of a burst-noise channel. Bell Syst. Tech. J. **39**(5), 1253–1265 (1960). https://doi.org/10.1002/j.1538-7305.1960.tb03959.x
8. Kozlov, A., Krouk, E., Ovchinnikov, A.: An approach to development of block-commutative codes with low density of parity check, vol. 8, pp. 9–14. Izvestiya vuzov, Priborostroenie (2013). https://doi.org/10.15217/issn1684-8853.2017.2.58
9. Krouk, E., Ovchinnikov, A.: Exact burst-correction capability of Gilbert codes. Informatsionno-upravliaiushchie sistemy **1**, 80–87 (2016). https://doi.org/10.15217/issn1684-8853.2016.1.80
10. Krouk, E., Ovchinnikov, A.: Code-based public-key cryptosystem based on bursts-correcting codes. In: AICT 2017: The Thirteenth Advanced International Conference on Telecommunications, pp. 90–92 (2017)
11. Krouk, E., Ovchinnikov, A.: 2-stripes block-circulant LDPC codes for single bursts correction. In: De Pietro, G., Gallo, L., Howlett, R.J., Jain, L.C. (eds.) Intelligent Interactive Multimedia Systems and Services 2016. SIST, vol. 55, pp. 11–23. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39345-2_2
12. MacKay, D.J.C., Neal, R.M.: Near Shannon limit performance of low density parity check codes. Electron. Lett. **33**(6), 457–458 (1997). https://doi.org/10.1049/el:19970362
13. Proakis, J., Salehi, M.: Digital Communications, vol. 5. McGraw-Hill, New York (2008)
14. Richardson, T.J., Urbanke, R.L.: The capacity of low-density parity-check codes under message-passing decoding. IEEE Trans. Inf. Theory **47**(2), 599–618 (2001). https://doi.org/10.1109/18.910577
15. Ryan, W., Lin, S.: Channel Codes: Classical and Modern. Cambridge University Press, Cambridge (2009)
16. Chung, S.-Y., Forney, G.D., Richardson, T.J., Urbanke, R.: On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit. IEEE Commun. Lett. **5**(2), 58–60 (2001). https://doi.org/10.1109/4234.905935
17. Veresova, A.M., Ovchinnikov, A.A.: About one algorithm for correcting bursts using block-permutation LDPC-codes. In: 2019 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF), pp. 1–4 (2019). https://doi.org/10.1109/WECONF.2019.8840580