European Conference on Optical Communication (ECOC) 2022 © Optica Publishing Group 2022

Low-Latency Low-Overhead Zipper Codes

Bashirreza Karimi [(1)], Masoud Barakatain [(1)], Yoones Hashemi [(1)], Deyuan Chang [(1)], Hamid Ebrahimzad[(1)], Chuandong Li [(1)]

[(1)] Huawei Technologies Co. Ltd, Ottawa, Canada, K2K 3J1, bashirreza.karimi@huawei.com

Abstract *A new hard-decision FEC scheme, suitable for high-throughput applications, is proposed that is based on zipper framework and is able to reduce the required memory and latency significantly compared to the conventional zipper codes.*

Introduction

High-speed optical transceivers operating at 800 Gbit/s and beyond have recently become commercially available. As in previous proposals (e.g., ITU-T G.709.2/Y.1331.2[1] and 400ZR[2]), forward error correction (FEC) is an essential component of these optical communication systems, enabling high-rate transmission. Due to complexity constraints at high data throughputs, FEC solutions with hard-decision decoding are preferred.

Staircase codes[3] are spatially-coupled FEC schemes based on algebraic BCH codes that have been included in several recommendations[1][2]. With iterative hard-decision decoding, staircase codes can perform very close to the capacity of the binary symmetric channel (BSC) with low decoding complexity. Not only these hard-decision FEC schemes are attractive as stand-alone FEC solutions in optical communications, but also in concatenated coded modulation schemes with bit interleaved coded modulation (BICM) and multilevel coding (MLC)[4][5]. In a concatenated scheme, typically an inner soft-decision code is concatenated with an outer hard-decision outer code. The inner code is tasked with reducing the bit error rate (BER) to a certain level, after which the outer code takes over and corrects the remaining errors down to below 10^{-15}.

Typically in BICM and MLC schemes, the best performance is obtained when the outer code overhead is ultra-low (around 1.5-2.5%). However, the required memory at the decoder grows rapidly as the overhead decreases. This issue was partially addressed with the introduction of zipper codes[6]. However, in many high-throughput practical applications, even the required memory of zipper codes is still too high and results in unacceptable latency. In this paper, we propose a FEC scheme based on the zipper framework that can operate with a decoding memory up to 13 times less than the conventional zipper frameworks and up to 26 times less than the staircase codes at low overheads. Proposed codes can operate within 0.6dB gap to the BSC channel capacity.

Zipper Code Framework

A zipper code framework is described by its three main components: a component code, a zipping pair, and an interleaver map. In the original work[4], a BCH code C of length n, dimenssion k, and correction capability t was considered as the component code. The zipper *buffer* is an infinite sequence of BCH codewords $C_0, C_1, C_2 \ldots$ that form a semi-infinite matrix with n columns. The buffer is divided into two sets: virtual buffer, A, and real buffer, B. Sets A and B are called a zipping pair (A, B). The bits of the virtual set are a direct copy of the bits in the real buffer through the interleaver map, φ. For practical purposes, the interleaver map is considered to be bijective, periodic and causal[6]. Let $m_i = |A_i|$ denote the length of the i-th row of the virtual set. The length of the corresponding real row therefore is $n - m_i$. Generally, m_i's for different rows can be different but we must have $m_{i+\vartheta} = m_i$, where $\vartheta > 0$ is the interleaver map period. It was shown that a diagonal interleaver map with $m_i = m = \frac{n}{2}$ in all rows, is able to provide very good performance[6]. Note that each bit is protected by 2 codes, once in the real buffer and once in the virtual buffer, but only the bits of the real buffer are transmitted over the communication channel.

A zipper code is decoded using a sliding-window decoding algorithm where M consecutive received rows are decoded iteratively using an algebraic BCH decoder[6]. This iterative decoder is invoked whenever a new row is received. In practice, to have a sharp waterfall performance, the ratio M/m should be large enough. To reduce the decoding complexity, the decoding subroutine is performed after receiving each μ rows, called a *chunk*. At the arrival of each chunk, the decoder outputs the oldest chunk within the window. In order to determine the decoder memory size for the window decoder, we first need to define a new parameter called *maximum lookback* denoted by λ. This parameter denotes the maximum number of older rows required in filling the virtual bits of each row. At the decoder, there exist Mm bits within the window decoder and an additional λm older bits have to be kept to

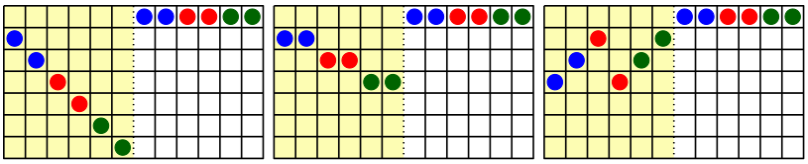

Fig. 1: Zipper code with $m = 6$ for diagonal (Left), quasi-diagonal type1 with $c = 2$ (middle), and quasi-diagonal type 2 (right) with $c = 2$ structures

perform the decoding. Thus, the total decoder memory size is $(\lambda + M) \times m$. As an example, the decoder memory sizes of a staircase code and a delayed diagonal zipper code is $(m + M) \times m$ and $(m/2 + M) \times m$, respectively.

Error floor in decoding various zipper frameworks (including staircase and diagonal zipper codes), is caused by certain structures called *stall patterns*. A stall pattern is a set of error locations in various rows of the real buffer B that cannot be corrected by the component BCH decoder which corrects up to t errors. Small stall patterns have the dominant impact on the error floor performance of zipper codes[6]. In the diagonal zipper code, the smallest stall patrtern is of size $(t + 1)(t + 2)/2$.

Low-Latency Zipper Codes

Although, comparing to staircase codes, diagonal zipper codes can decrease the required memory size for encoding and decoding by about a factor of 2, their latency for ultra-low overheads is still too high and thus not acceptable in high-throughput applications. The interleaver map φ is an important component of the zipper framework that directly affects the code performance and memory requirement. To deal with memory and latency issue in conventional zipper frameworks, here, we propose a new interleaver map bit protection scheme that can significantly reduce the memory size while maintaining a good FEC performance.

1. Quasi-Diagonal Zipper Framework

In the diagonal zipper codes, the virtual bits of each row are obtained from the real bits of m previous rows. Equivalently, the real bits of each row are distributed among the next m rows. This induces a high dependency between the rows of real and virtual buffers. Here, to achieve a sharp waterfall performance a large decoding window size, M, as a multiple of m, is required.

In order to reduce the dependency between the rows of real and virtual buffers, we introduce a quasi-diagonal interleaver map with the *coupling factor* c. This interleaver map allows up to c bits of the virtual part of a row be copies of real bits of a single row, thereby reducing the dependency by a factor of c. Fig. 1 shows a zipper code with $m = 6$ and three types of interleaver maps: one diagonal (left), and two quasi-diagonals with $c = 2$ (middle and right). As shown in Fig. 1, for the diagonal design, the maximum lookback is $\lambda = 6$ while for the quasi-diagonal designs, $\lambda = 3$.

It is easy to see that with quasi-diagonal interleaving, the required encoder memory size is $\frac{m(m/c+1)}{2} + m$ bits and the required decoder memory is $m(m/(2c) + M)$ bits, but here M has to be measured against m/c. Therefore, increasing the coupling factor, c, reduces the required memory sizes considerably. Note that the decoding latency is linearly proportional to mM and therefore decreases significantly with increasing c.

Increasing c, however, may induce an early error floor to the BER curve of the quasi-diagonal zipper code. Error floor in decoding various zipper frameworks (including staircase, diagonal and quasi-diagonal zipper codes), is caused by certain structures called *stall patterns*. A stall pattern is a set of error locations in various rows of the real buffer B that cannot be corrected by the component BCH decoder which corrects up to t errors. Small stall patterns have the dominant impact on the error floor performance of zipper codes[6]. The quasi-diagonal interleaving reduces the size of smallest stall pattern and also increases the multiplicity of the stall patterns. For example, when $c > t$, the minimum-size stall patterns are of size $t + 1$, where as in the diagonal interleaving, the smallest stall patrtern is of size $(t + 1)(t + 2)/2$. Next, we propose an approach to address this problem, leading to a powerful zipper code design with low latency and good waterfall and error floor performances

2. Extra Level of Protection

Where a stall pattern is formed, errors cannot be resolved by the component codes that protect the bits in error. Adding protections by additional component codes can potentially resolve the stall pattern errors. Thus, we propose a new zipper framework in which each bit is protected by three component codes. This extra level of protection can be implemented by considering the zipping triplet (A_1, A_2, B) with two virtual sets, A_1 and A_2, and their corresponding (and possibly different) interleaver maps denoted by (φ_1, φ_2) with

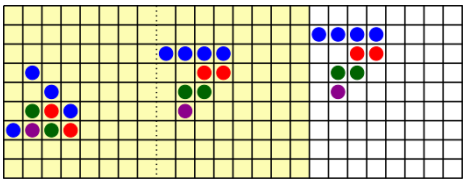

Fig. 2: An example of stall pattern of size 9 for proposed zipper code with $m = 8$ and coupling factors $c_1 = c_2 = 4$.

coupling factors (c_1, c_2), respectively. In order to increase the size of minimum stall patterns, we design (φ_1, φ_2) interleavers such that each bit in the real buffer is mapped to different rows of the two virtual buffers. For example, one can consider the type 1 and type 2 interleaver maps of Fig. 1 for each of the virtual buffers. Note that because of this extra level of protection, one might need to use a BCH component code with a larger Galois Field (GF) size.

Based on the discussion of [6], the size of smallest stall pattern is related to the number of affected rows by that pattern. In a conventional zipper code with a diagonal interleaver, if there exists $t+1$ errors in i-th row of the real buffer, these errors are distributed in $t+1$ distinct rows of the virtual buffer. As a result, the decoder is trapped in a stall pattern if all $t+2$ rows (including the i-th row) have at least $t+1$ errors. The size of the smallest stall pattern is therefore $(t+1)(t+2)/2$, whcih is proportinal to the number of affected rows, $t+2$. In the proposed low-latency zipper framework with $c > t$, since we have two virtual buffers with different interleaver maps, the number of affeccted rows is at least $t+3$. Therefore, the size of the minimal stall pattern for this framework scheme is lower bounded by $(t+1)(t+3)/3$. For example, when BCH code with triple-error-correction capability is used, the smallest stall pattern size larger than or equal to 8. Fig. 2 shows an example of stall patterns with minimum numbers of affected rows for $t = 3$ and size 9. Note that while the minimum stall pattern size is smaller than that in the conventional scheme, with this extra level of protection we did not observe error floor in the region of interest for the proposed codes.

Simulation Results

In this section, we present our simulation results for different types of zipper codes including, diagonal (D), quasi-diagonal (QD), and quasi-diagonal with extra level of protection (QDE). Here, we consider triple-error-correction BCH as the component codes in the zipper framework design for three low overheads: 1.65%, 2.04%, and 2.61%. In simulations, we have considered BCH component codes generated from GF size

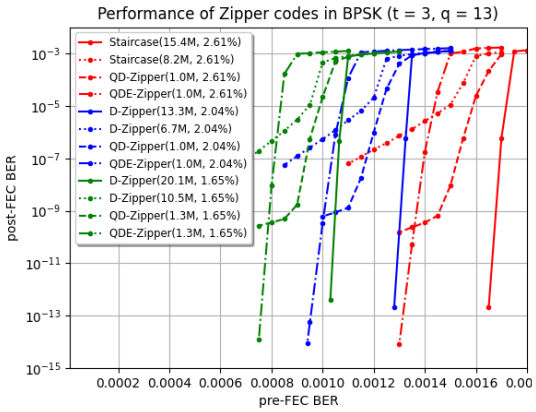

Fig. 3: Performance curves of, staircase, diagonal, and proposed zipper codes for different low overheads.

2^{13}. In Fig. 3, we plot the post-FEC BER versus pre-FEC BER performance curves for various code designs. For QDE-zipper, code parameters (M, m, c) are $(2400,400,8)$, $(1950,400,8)$, and $(1536,650,5)$ for OHs from 1.65-2.61%, respectively. It can be observed from these curves that, compared to the diagonal interlaver, the quasi-diagonal interleaver map is able to reduce the memory size at the cost of an early error floor. The designed QDE zipper codes, on the other hand, can provide a waterfall performance as sharp as the diagonal zipper codes, without any sign of an error floor down to a 10^{-14} BER, while keeping memory size significantly lower. All of the proposed zipper codes operate within 0.6dB gap to the BSC channel capacity and can also be implemented using the more practical pipeline decoding with minimal performance loss. Our simulation results in coded modulation schemes show that the performance loss of the proposed scheme is less than 0.03dB compared to conventional staircase and zipper codes.

Conclusions

In this paper, we proposed a new hard-decision FEC scheme based on zipper framework, named QDE-zipper code, which is able to reduce the required memory and latency for low-overhead high-throughput applications. Compared to the existing diagonal zipper and staircase codes with low-overheads, QDE-zipper code was shown to reduce the decoding memory by up to a factor of 13 and 26, respectively. The proposed scheme attains a sharp waterfall performance without any sign of error floor down to 10^{-14} and operates within 0.6 dB gap to the BSC channel limit.

Acknowledgements

The authors would like to thank Alvin Y. Sukmadji for his helpful comments.

References

[1] ITU-T Recommendation G.975.1/Y.1331.2, "Long-reach interface," Tech. Rep., 2018.

[2] B. Smith, I. Lyubomirsky, and S. Bhoja. (2017) *Leveraging 400G ZR FEC technology*. IEEE 802.3 Beyond 10km Optical PHYs Study Group. [Online]. Available:http://www.ieee802.org/3/B10K/public/1711/lyubomirsky_b10k_01_1117.pdf

[3] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, "*Staircase codes: FEC for 100 Gb/s OTN*," J. Lightw. Technol., vol. 30, no. 1, pp. 110–117, 2012. DOI: 10.1109/JLT.2011.2175479

[4] M. Barakatain, and F.R. Kschischang, "*Low-complexity concatenated LDPC-staircase codes*," J. Lightw. Technol., vol. 36, no. 16, pp. 2443–2449, 2018, (correction: vol. 37, no. 3, p. 1070, 2019). DOI: 10.1109/JLT.2018.2812738

[5] M. Barakatain, D. Lentner, G. Böecherer, and F.R. Kschischang, "*Performance-complexity tradeoffs of concatenated FEC for higher-order modulation*," J. Lightw. Technol., vol. 38, no. 18, pp. 2944–2953, 2020. DOI: 10.1109/JLT.2020.2983912

[6] A. Y. Sukmadji, U. Martínez-Peñas, and F. R. Kschischang, "*Zipper codes: Spatially-coupled product-like codes with iterative algebraic decoding*," in Proc. Can. Workshop Inf. Theory, Jun. 2019, pp. 1–6. DOI: 10.1109/CWIT.2019.8929906