Efficient Implementation of Parallel Concatenated Gallager Codes with Single Encoder

Aswathy G P
Department of Electronics and Communication Engineering
Mar Baselios College of Engineering and Technology
Thiruvananthapuram, India
aswathygp19@gmail.com

Niyas K Haneefa
Department of Electronics and Communication Engineering
Mar Baselios College of Engineering and Technology
Thiruvananthapuram, India
niyaskhaneefa@gmail.com

***Abstract*—Parallel Concatenated Gallager Codes (PCGCs) have proved to be an effective way to improve the bit error rate (BER) performance by increasing the redundancy at the same time reducing the encoding and decoding complexity when compared to dedicated Low Density Parity Check (LDPC) code of same length and code rate. Existing PCGC with single encoder has not been extensively successful due to its performance limitation, decoder complexity and decoding delay. In this paper we propose an efficient decoding methodology for PCGC with single encoder. We show that the proposed methodology outperforms existing PCGC with single encoder in terms of BER performance in AWGN channel. We also present that the proposed decoder is much less complex than existing one.**

***Index Terms*—LDPC, Parallel concatenation, PCGC, Decoding complexity.**

I. Introduction

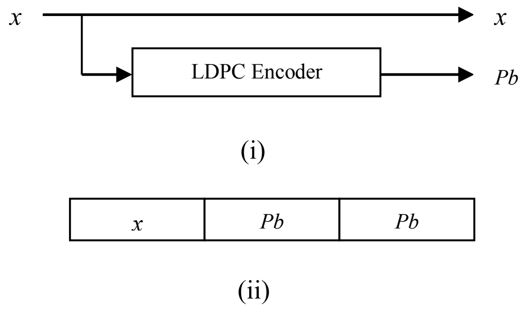
Low-density parity-check (LDPC) codes are today's most widely used Forward Error Correction (FEC) codes. It is a class of linear block codes. These codes were first introduced by Robert Gallager in his PhD Thesis in 1960 [1]. But these codes were not deployed much as they were thought to be impractical due to its implementation complexity at that time. After the discovery of Turbo codes by Berrou in 1993, LDPC codes were re-discovered by D. MacKay and R. Neal in 1995 [2], [3]. LDPC codes are well known for their capacity approaching performance which increases with increase in block length.

Fully parallel LDPC decoder for longer block length, usually achieves very good performance. But they suffers from prohibitive implementation complexity due to widespread connectivity in the bipartite Tanner graph. Especially the routing overhead with a large number of global routing wires is extremely difficult for hardware implementation. Concatenation of codes is usually done to attain the error correction capability of longer codes, with reduced decoding complexity. Parallel concatenation has become one of the fieriest topics in coding theory after the discovery of Turbo codes [4]. Eventually H. Behairy and S. C. Chang introduced Parallel Concatenated Gallager Codes (PCGCs) in [5]. Conventional PCGC [5], [6], [7], [9] is a class of concatenated codes in which two component LDPC codes are connected in parallel using interleaver free Turbo code architecture. But there are some drawbacks for Conventional PCGC. It fails to achieve the entire coding gain provided by concatenation of two component LDPC codes. Since serial decoding is used, decoding delay is more. Subsequent attempts of parallel concatenation of LDPC codes are presented in [10], [11], [12] and [13] which have not been widely successful due to their limitations.

The remaining sections of this paper are structured as follows: In Section II we describe existing PCGC with single encoder and mention its limitations. In Section III we present the proposed decoding methodology for PCGC with single encoder and specify the advantages of proposed approach over existing scheme. Section IV presents the simulation results for the bit error rate (BER) performances of proposed PCGC with single encoder in Additive White Gaussian Noise (AWGN) channel and section V concludes the paper.

II. Existing PCGC with Single Encoder

In PCGC with single encoder [13], [14] a single LDPC code with Mean Column Weight (MCW) greater than 2.5 is used as the component code. Fig. 1. shows the encoder and concatenated codeword structure for PCGC with single encoder. Here *x* represents the systematic information bits and *Pb* represents the parity bits produced by the component encoder. Here the same parity bits are transmitted twice.



(i)

(ii)

Fig. 1. (i) Encoder (ii) Concatenated codeword structure for PCGC with single encoder [13].

The decoder for PCGC with single encoder consists of two simple LDPC decoders which are defined for the same parity check matrix. This is in contrast to other PCGCs explained in [5], [8], [10], where the component decoders were defined for different parity check matrices. The component decoders are connected in parallel using a set of check nodes called interconnecting check nodes. Each interconnecting check node corresponds to a parity check equation of the parity check matrix. Bit nodes of each component decoder are connected to interconnecting check nodes if they are involved in the corresponding parity check equations [13].

Decoding of PCGC with single encoder is an iterative process. A local iteration is referred to one complete cycle of the message passing algorithm, performed by the component LDPC decoder. Fig. 2. shows the existing decoder architecture for PCGC with single encoder.

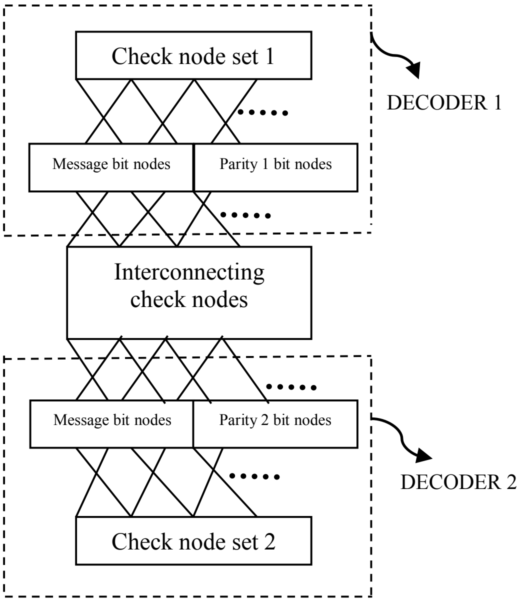


Fig. 2. Existing decoder for PCGC with single encoder [13], [16].

A super iteration for PCGC with single encoder can be explained as follows [13], [16]. In the first step of super iteration, both component decoders will simultaneously and independently perform one local iteration of the iterative decoding i.e. the intrinsic information from the bit nodes are transferred to the check nodes of the corresponding decoder and bit nodes are updated based on the extrinsic messages available from the check nodes. In the second step, each bit node of first component decoder is updated (i.e. log likelihood ratio (LLR) value is changed) by using the updated intrinsic information of the bit nodes of second component decoder and similarly each bit node of second component decoder is updated using the modified intrinsic information of bit nodes of first component decoder through the interconnecting check nodes. This process of super iteration continues until one of the decoder converges to a valid codeword or a maximum number of super iterations are over. In the latter case, the output from the second component decoder is declared as the best estimate of the transmitted sequence as the super iteration process ends here.

The main drawback of PCGC with single encoder is the complexity associated with its decoder due to interconnecting check nodes. Since PCGC decoder has got a symmetric structure, it is suggested in [13] to make use of this property by using additional registers in each bit node to reduce its complexity. But decoding delay is increased by a factor of $\frac{1}{3}$ with reduction in decoder complexity. Also each super iteration of PCGC with single encoder is a time-consuming process, which also accounts for decoding delay. Another demerit of PCGC with single encoder is that its performance is not as good as a single dedicated LDPC of same frame length and code rate [14].

III. Proposed Decoder for PCGC with Single Encoder

Fig. 3. shows the proposed decoder architecture for PCGC with single encoder. Two simple LDPC decoders are used as component decoders. They are defined for the same parity check matrix. Both the decoders are connected in parallel without any set of interconnecting check nodes, hence reducing the decoder complexity.

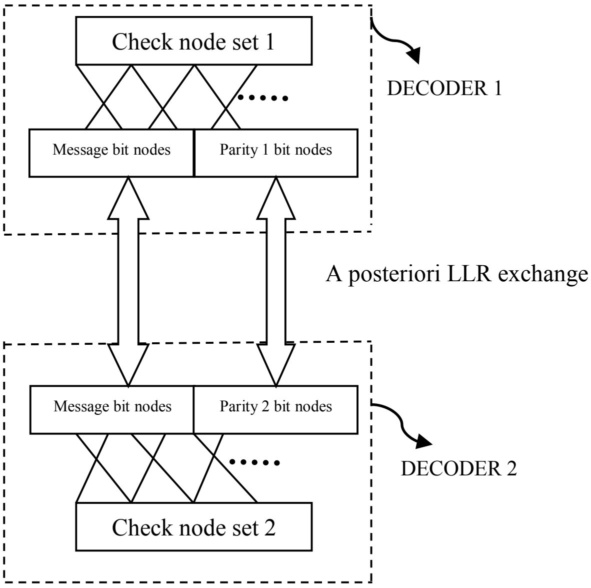


Fig. 3. Proposed decoder for PCGC with single encoder.

During a super iteration, each component decoder simultaneously and independently performs a local iteration to obtain the corresponding *a posteriori* LLRs. After every local iteration, each component decoder exchange their *a posteriori* LLRs and this is used by the other component decoder as *a priori* for the next super iteration. Similar principle is used in [8], where only the extrinsic information corresponding to systematic information bits are used as *a priori* by the other component decoders, since the parity sets were different. Also during each super iteration, component decoders performed a number of local iterations before passing any information to other component decoders. This increased the decoding delay.

In PCGC with single encoder, both decoders are searching for the same codeword (since parity sets transmitted are same). So here, *a posteriori* LLR (*a priori* information + channel information + extrinsic information) of whole codeword (systematic information bits and parity bits) is exchanged by each component decoder and is used as *a priori* information by the other component decoder in subsequent super iteration. On the transmitting side, the same parity sets are sent twice. With this setting, there is no need to avoid the channel information from the *a posteriori* metrics, to be shared between the component decoders. This is because the parity set received by any one of the component decoder may be different and independent from the parity set received by the other component decoder. Exchange of *a posteriori* LLRs after every local iteration helps the component decoders to use channel information of both the parity sets effectively. Also inorder to reduce the decoding delay, only a single local iteration is performed by each component decoder during every super iteration.

Let c = (c_1 , c_2 , …c_L) be the transmitted concatenated codeword and r = (r_1 , r_2 , …r_L) denotes the received concatenated codeword at the channel output. The PCGC decoder first de-multiplex the received sequence into two vectors to distribute them for component decoders. Let y_x^0 denotes the received sequence corresponding to systematic information bits *x*, while y_p^1 and y_p^2 represents the received sequence corresponding to first and second repetition of parity set *Pb*. De-multiplexed vectors for decoder 1 and decoder 2 are $y^1 = [y_x^0 , y_p^1]$ and $y^2 = [y_x^0 , y_p^2]$ respectively. Let Y^1 and Y^2 represents the LLRs of each bits corresponding to y^1 and y^2 respectively.

Let us define super iteration for proposed decoder by considering Minimum Sum Algorithm (MSA) as iterative decoding algorithm [15].

1. Initially the bit nodes of component decoders are loaded with LLR values (Y^1 and Y^2) as intrinsic information.
2. Now decoder 1 and decoder 2 will independently and simultaneously perform one local iteration of MSA and obtain the *a posteriori* LLRs. Let us explain this step in detail [13].

   Let L_m^q be the intrinsic information associated with the m^{th} bit node of q^{th} decoder. Bit node *m* of decoder *q* will send message ($M^q_{m\to n}$) to check node *n* of decoder *q*. This is given by,

   $$M^q_{m\to n} = L^q_m - E^q_{n\to m} \qquad (1)$$

   where, $E^q_{n\to m}$ represents the extrinsic information sent from check node *n* of decoder *q* to bit node *m* of decoder *q* during previous super iteration ($E^q_{n\to m} = 0$ for initial super iteration). Once check nodes receive the messages, they will forward extrinsic information ($E^q_{n\to m}$) towards neighbouring bit nodes of decoder *q*, which is given by,

   $$E^q_{n\to m} = \prod_{\boldsymbol{m'} \epsilon B^q_n, m' \neq m} \text{sign}\,(M^q_{m'\to n}) \min|\, M^q_{m'\to n}| \quad (2)$$

   where B^q_n represents the set of bit nodes connected to check node *n* of decoder *q*. Now the LLR values of bit nodes of both decoders are updated.

   $$L^q_m = Y^q_m + \sum_{n \epsilon A^q_m} E^q_{n\to m} \qquad (3)$$

   where A^q_mrepresents the set of check nodes connected to bit node *m* of decoder *q*.
3. Check for syndrome condition. If Syndrome = 0 for either of the component decoders, then stop decoding.
4. If Syndrome ≠ 0, increment the super iteration count and exchange the *a posteriori* LLRs of each component decoder.
5. Exchanged *a posteriori* LLRs are used as *a priori* information by each component decoder in subsequent super iteration.

Repeat the super iteration process until a valid codeword is obtained by any one of the component decoders or super iteration count reaches its maximum value. If the component decoders does not converge to any valid codeword, the output from the second component decoder is considered as the transmitted codeword estimate, as the super iteration process ends here.

Proposed decoding algorithm for PCGC with single encoder is given by,

```
procedure (start)
  ➢ Load LLR to both component decoders
    for $q$ = 1 : 2
      for $m$ = 1 : $N$
        $L_m^q = Y_m^q$
      end
    end
  ➢ Initialize Super iteration count
    $SI$ = 0
  ➢ Assign all $E^q_{n\to m}$ to 0 for initial super iteration
    for $q$ = 1 : 2
      for $m$ = 1 : N
        for $n$ = 1 : $M$
          $E^q_{n\to m}$= 0
        end
      end
    end

repeat steps
  ➢ Intrinsic message transfer
    for $q$ = 1 : 2
      for $m$ = 1 : $N$
        for $n \epsilon A_m^q$
          $M^q_{m\to n} = L^q_m - E^q_{n\to m}$
        end
      end
    end
  ➢ Extrinsic information transfer
    for $q$ = 1 : 2
      for $n$ = 1 : $M$
        for $m \epsilon B_n^q$
```

```
            $E_{n\to m}^{q} = \prod_{m' \in B_{n}^{q}, m' \neq m} \operatorname{sign}(M_{m'\to n}^{q}) \min |M_{m'\to n}^{q}|$
          end
        end
      end
  ➢ Update decoder bit nodes
      for $q = 1 : 2$
        for $m = 1 : N$
            $L_{m}^{q} = Y_{m}^{q} + \sum_{n \in A_{m}^{q}} E_{n\to m}^{q}$
            $\hat{c}_{m}^{q} = \begin{cases} 1, & L_{m}^{q} < 0 \\ 0, & L_{m}^{q} \geq 0 \end{cases}$
        end
      end
  ➢ Check syndrome or maximum super iterations
        if $H(c^{1})^{T} = 0$ (or) $H(c^{2})^{T} = 0$ (or) $SI = SI_{max}$
          Stop iteration
        end
```

Until here, the process is similar to existing PCGC with single encoder [13]. After performing a local iteration, the intrinsic information in bit nodes of each component decoder is updated by adding the total *a posteriori* information obtained from the bit nodes of other component decoder.

```
        $SI = SI + 1$
  ➢ Exchange and addition of a posteriori LLRs in bit
    nodes of both decoders
        for $q = 1 : 2$
          for $m = 1 : N$
              $Y_{m}^{q} = Y_{m}^{q} + L_{m}^{q'}$, where $q' = \begin{cases} 1, & q = 2 \\ 2, & q = 1 \end{cases}$
          end
        end
      repeat until convergence (or) $SI = SI_{max}$
procedure (stop)
```

In proposed PCGC with single encoder, exchange of *a posteriori* LLRs increases the reliability of each codeword bit after every super iteration. In this new methodology also, decoder 1 and decoder 2 performs local iteration simultaneously and every local iteration is followed by an exchange of information (*a posteriori* LLR) by the component decoders. So convergence is much quicker for proposed PCGC.

In proposed decoder, interconnecting check nodes are avoided. Thus if a parity check matrix of dimension M x N is used, then there is a reduction of M check node units compared to existing PCGC decoder. Hence decoder complexity is greatly reduced. Since additional computations performed by these interconnecting check nodes are also avoided, decoding delay is reduced significantly. As proposed in [13], since the PCGC decoder is having a symmetric structure, we can make use of this to reduce the implementation complexity. This can be done by replacing the PCGC decoder with a single LDPC decoder (defined for same parity check matrix) with some additional registers in each bit node. However, decoding delay is increased by a factor of $\frac{1}{2}$ with reduction in decoder complexity. But the decoding delay for proposed decoder is much less compared to existing decoder for PCGC with single encoder. Also the performance of the proposed PCGC with single encoder is almost similar to a dedicated LDPC of same rate and frame length. But proposed PCGC requires more number of decoding iterations compared to dedicated LDPC inorder to achieve the same performance.

IV. Simulation Results

The simulation results for the BER performance of rate $\frac{1}{3}$ existing PCGC with single encoder and proposed PCGC with single encoder for frame length 600 are presented below. Component code used for both are rate $\frac{1}{2}$ LDPC with dimension 200 x 400 and MCW 3 as specified in [13].

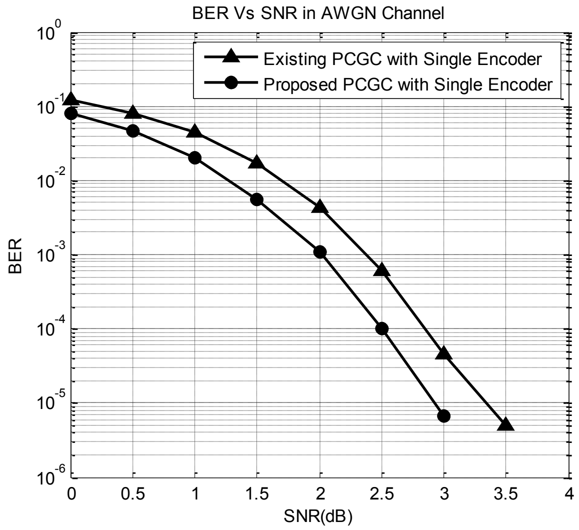


Fig. 4. BER comparison of Existing PCGC and Proposed PCGC with single encoder in AWGN channel

For existing PCGC with single encoder, the maximum number of super iteration was set to 25. For proposed PCGC, the maximum number of super iteration was set to 25. For simulations, we used MSA in AWGN channel. In Fig. 4 we compared the BER performance of existing PCGC with single encoder and proposed PCGC with single encoder in AWGN channel. The performance of proposed PCGC with single encoder is much better than existing PCGC with single encoder in all SNR regions.

In Fig. 5, the simulation results for the BER comparison of rate $\frac{1}{3}$ existing PCGC with single encoder, proposed PCGC with single encoder and dedicated LDPC of frame length 600 are presented. For both existing PCGC and proposed PCGC with single encoder, the maximum number of super iteration was set to 25. For dedicated LDPC, the maximum number of decoding iterations (for MSA) was set to 5. We see that proposed PCGC almost achieves the same performance as dedicated LDPC (of same frame length and code rate) in all SNR regions.

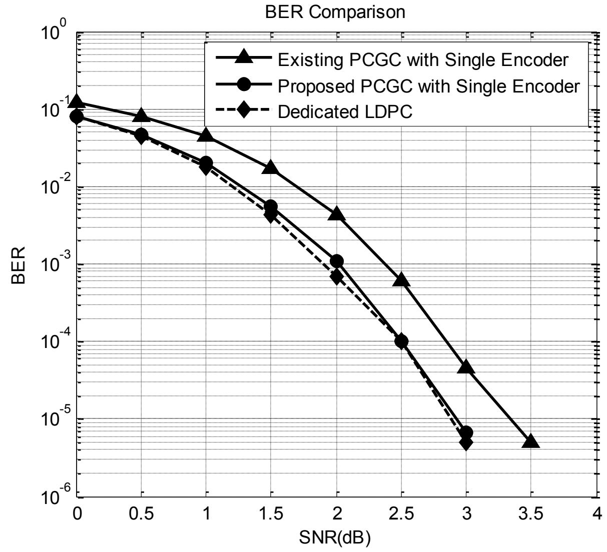


Fig. 5. BER comparison of Existing PCGC with single encoder, Proposed PCGC with single encoder and Dedicated LDPC in AWGN channel

V. Conclusion

In this paper, we have proposed a novel decoding methodology for PCGC with single encoder which is effective than the existing methodology. We have shown that the proposed PCGC outperforms existing one in terms of decoder complexity, BER performance and decoding delay. We have also shown that performance of proposed PCGC with single encoder is almost comparable to dedicated LDPC of same rate and frame length. As future work, we suggest to find an efficient PCGC with performance better than dedicated LDPC.

References

[1] R. Gallager, "Low-Density Parity-Check Codes", *IRE Transaction on Information Theory*, vol. IT-8, pp. 21–28, January 1962.

[2] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes", *Electronics Lett*ers, vol.32, pp.1645-1646, 1996.

[3] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes", *Electronics Letters (reprint with printing errors corrected)*, vol.33, pp.457-458, March 1997.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes", *Proceedings of IEEE International Conference on Communication*, pp.1064-1070, May 1993.

[5] H. Behairy and S. C. Chang, "Parallel Concatenated Gallager codes", *Electronics Letters*, vol. 36, no. 24, pp. 2025-2026, November 2000.

[6] H. Behairy and S. C. Chang, "Parallel Concatenated Gallager Codes for CDMA Applications", *Proceedings of IEEE GLOBECOM'01*, San Antonio, TX, pp. 1002-6, 2001.

[7] H. Behairy and S. C. Chang, "Analysis and design of parallel concatenated Gallager codes", *Electronics Letters*, vol. 38, no. 18, pp. 1039- 1040, Aug 2000.

[8] Hatim M. Behairy and Mohammed Benaissa, "Multiple Parallel Concatenated Gallager Codes: Code Design and Decoding Techniques", *IETE Journal of Research*, vol: 59, issue: 6, page no: 659-664, Nov-Dec 2013.

[9] H. M. Behairy, "Dual Mode PCGCs for Advanced Wireless Communications Networks", *Proceedings of IEEE International Design and Test Workshop*, IDT 2007, Cairo, Egypt, Dec.2007.

[10] P. C. Catherine and K. M. S. Soyjaudah, "Parallel Concatenation of LDPC codes with Two Sets of Source Bits", *IEEE Proceedings of International Conference on Emerging Trends in Engineering & Technology*, page no: 112-115, 2011.

[11] J. C. Serrato and T. O'Farrell, "Parallel Concatenated Gallager Codes using Euclidean and Projective Geometry LDPC Codes", *Proceedings of London Communications Symposium (LCS 2004)*, University College London, UK, September 2004.

[12] J. C. Serrato and T. O'Farrell, "Structured Parallel Concatenated LDPC codes", *Proceedings of London Communications Symposium (LCS 2006)*, University College London, UK, September 2006.

[13] R. Pravin Kumar and Rakhesh Singh Kshetrimayum, "An Efficient Methodology for Parallel Concatenation of LDPC codes with reduced complexity and decoding delay", *IEEE Proceedings of the National Conference on Communications (NCC)*, page no: 1-5, February 2013.

[14] R. Pravin Kumar, "Parallel Concatenation of LDPC codes and its applications", M. Tech. thesis, Department of Electronics and Electrical Engineering, IIT Guwahati, India, April 2013.

[15] Sarah J. Johnson, Iterative Error Correction, Cambridge University Press, 2010.

[16] Aswathy G. P and Niyas K. Haneefa, "A Survey on Methods of Parallel Concatenation of LDPC Codes", *International Journal of Engineering and Advanced Technology,* vol. 5, issue: 4, page no: 124 -129, April 2016.