ISSN 2319-2720
Volume 7, No.2, April – June 2018
International Journal of Computing, Communications and Networking
Available Online at http://www.warse.org/ijccn/static/pdf/file/ijccn25722018.pdf
https://doi.org/10.30534/ijccn/2018/25722018

A NEW APPROACH FOR ERROR CORRECTION DECODING BASED ON THE BP ALGORITHM IN LTE AND WIMAX SYSTEMS

Poonam Kashyap[1], R. Harshitha priyanka[2] ,Poornima s yadav[3] ,Prof. Shylaja B R[4]
[1]Student, EWIT, India, poonamknov1996@gmail.com
[2]Student, EWIT,India, harshithapriyanka1711@gmail.
[3]Student, EWIT, India, poornimsyadav@gmail.com
[4]Assistant Professor, EWIT, India, shylaja.b.r@gmail.com

ABSTRACT

Abstract—Many wireless communication systems such as IS-54, enhanced data rates for the GSM evolution (EDGE), worldwide interoperability for microwave access (WiMAX) and long term evolution (LTE) have adopted low-density parity check (LDPC), tail-biting Convolutional, and turbo codes as the forward error correcting codes (FEC) scheme for data and overhead channels. Therefore, many efficient algorithms have been proposed for decoding these codes. However, the different decoding approaches for these two families of codes usually lead to different hardware architectures. Since these codes work side by side in these new wireless systems, it is a good idea to introduce a universal decoder to handle these two families of codes. The present work exploits the parity-check matrix (H) representation of tail biting Convolution and turbo codes, thus enabling decoding via a unified belief propagation (BP) algorithm. Indeed, the BP algorithm provides a highly effective general methodology for devising low-complexity iterative decoding algorithms for all Convolution code classes as well as turbo codes. While a small performance loss is observed when decoding turbo codes with BP instead of MAP, this is offset by the lower complexity of the BP algorithm and the inherent advantage of a unified decoding architecture.

Key words: Zigbee, GPS, GSM, Microcontroller, Wireless network, Fishermen.

1. INTRODUCTION

Until Recently, Most Known Decoding Algorithms For Convolution Codes Were Based On Either Algebraically Calculating The Error Pattern Or On Trellis Graphical Representations Such As In The MAP And Viterbi Algorithms. With The Advent Of Turbo Coding [1], A Third Decoding Principle Has Appeared: Iterative Decoding. Iterative Decoding Was Also Introduced In Tanner's Pioneering Work [2], Which Is A General Framework Based On Bipartite Graphs For The description of LDPC codes and their decoding via the belief propagation (BP) algorithm. In many respects, convolution codes are similar to block codes.

For example, if we truncate the trellis by which a convolution code is represented, a block code whose code words correspond to all trellis paths to the truncation depth is created. However, this truncation causes a problem in error performance, since the last bits lack error protection. The conventional solution to this problem is to encode a fixed number of message blocks L followed by m additional all-zero blocks, where m is the constraint length of the convolution code [4]. This method provides uniform error protection for all information digits, but causes a rate reduction for the block code as compared to the Convolution code by the multiplicative factor L/(L+m). In the tail-biting convolution code, zero-tail bits are not needed and replaced by payload bits resulting in no rate loss due to the tails. Therefore, the spectral efficiency of the channel code is improved. Due to the advantages of the tail-biting method over the zero-tail, it has been adopted as the FEC in addition to the turbo code for data and overhead channels in many wireless communications systems such as IS-54, EDGE, WiMAX and LTE [5, 6, 7].

Both turbo and LDPC codes have been extensively studied for more than fifteen years. However, the formal relationship between these two classes of codes remained unclear until Mackay in [8] claimed that turbo codes are LDPC codes. Also, Wiberg in [9] marked another attempt to relate these two classes of codes together by developing a unified factor graph representation for these two families of codes. In [10],McEliece showed that their decoding algorithms fall intothe same category as BP on the Bayesian belief network.Finally, Colavolpe [11] was able to demonstrate the use of theBP algorithm to decode convolutional and turbo codes.

The operation in [11] is limited to specific classes of Convolutional codes, such as convolutional self orthogonal codes (CSOCs).Also, the turbo codes therein are based on the serial structure while the parallel structure is more prevalent in practical application. .In LTE and WiMAX systems, the proposed decoders fortail-biting convolutional codes and turbo codes are basedon the Viterbi and MAP algorithms, respectively. However,many other efficient algorithms have been proposed to decodetail-biting convolutional codes as well as turbo codes. Forexample, in [3], the reduced

complexity wrap-around Viterbialgorithm was proposed to decode tail-biting convolutionalcodes in the WiMAX system to reduce the average numberof decoding iterations and memory usage. In addition,other decoding algorithms such as double traceback andbidirectional Viterbi algorithms were also proposed fortail-biting convolutional codes in LTE [4]. Finally in [5], the design and optimization of low-complexity high performance rate-matching algorithms based on circular buffers for LTE turbo codes was investigated. In this paper, we focus on the direct application of the BP algorithm used for LDPC codes to decode the tail-biting convolutional codes and turbo codes in WiMAX and LTE systems, respectively. Based on that, we propose a decoder with drastically lower implementation complexity than that proposed in the latest releases for these systems [5-7].

The rest of this paper is organized as follows. In Section II and III, the graphical representation of the tail-biting convolutional and turbo codes with the necessary notations and definitions used throughout this paper are introduced, followed by an investigation of the coding structures in WiMAX and LTE systems in Section IV. In Section V, simulation results for the performance of tail-biting convolutional and turbo codes using the proposed algorithm are introduced followed in Section VI by a complexity comparison between the proposed algorithm and the traditional ones. Finally, the paper is concluded in Section VII.

Existing System

For analysis purposes the packet-loss process resulting from the single-multiplexer model was assumed to be independent and, consequently, the simulation results provided show that this simplified analysis considerably overestimates the performance of regenerative error correction.Evaluation of regenerative error correction performance in multiple session was more complex in existing applications.Surprisingly, all numerical results given indicates that the resulting residual packet-loss rates with coding are always greater than without coding, i.e., regenerative error correction is ineffective in this application.

The increase in the redundant packets added to the data will increase the performance, but it will also make the data large and it will also lead to increase in data loss.

2. LITERATURE SURVEY

Forward Error Correction:

FEC is accomplished by adding redundancy to the transmitted information using a predetermined algorithm. Each redundant bit is invariably a complex function of many original information bits. The original information may or may not appear in the encoded output; codes that include the unmodified input in the output are systematic, while those that do not are nonsystematic.

An extremely simple example would be an analog to digital converter that samples three bits of signal strength data for every bit of transmitted data. If the three samples are mostly all zero, the transmitted bit was probably a zero, and if three samples are mostly all one, the transmitted bit was probably a one. The simplest example of error correction is for the receiver to assume the correct output is given by the most frequently occurring value in each group of three.

Table 1. FEC codes

Triplet received	Interpreted as
000	0
_00	0
0_0	0
00_	0
0__	0
_0_	0
0__	0
111	1
_11	1
1_1	1
11_	1
1__	1
_1_	1
__1	1

This allows an error in any one of the three samples to be corrected by "democratic voting". This is a highly inefficient FEC, and in practice would not work very well, but it does illustrate the principle.

In practice, FEC codes typically examine the last several dozen, or even the last several hundred, previously received bits to determine how to decode the current small handful of bits (typically in groups of 2 to 8 bits).Such triple modular redundancy, the simplest form of forward error correction, is widely used.Forward Error Correction (FEC) is a type of error correction, which improves on simple error detection schemes by enabling the receiver to correct errors once they are detected. This reduces the need for retransmissions. FEC works by adding check bits to the outgoing data stream. Adding more check bits reduces the amount of available bandwidth, but also enables the receiver to correct for more errors.Forward Error Correction is particularly well suited for satellite transmissions, where bandwidth is reasonable but latency is significant.

3. SYSTEM DESIGN

The two main categories of FEC are block coding and convolutional coding.

- Block codes work on fixed-size blocks (packets) of bits or symbols of predetermined size.

- Convolutional codes work on bit or symbol streams of arbitrary length.
- A convolutional code can be turned into a block code, if desired.
- Convolutional codes are most often decoded with the Viterbi algorithm, though other algorithms are sometimes used.

There are many types of block codes, but the most notable is Reed-Solomon coding because of its widespread use on the Compact disc, the DVD, and in computer hard drives. Golay, BCH and Hamming codes are other examples of block codes. Hamming ECC is commonly used to correct NAND flash memory errors. This provides single-bit error correction and 2-bit error detection. Hamming codes are only suitable for more reliable single level cell (SLC) NAND. Denser multi level cell (MLC) NAND requires stronger multi-bit correcting ECC such as BCH or Reed-Solomon. Nearly all block codes apply the algebraic properties of finite fields.

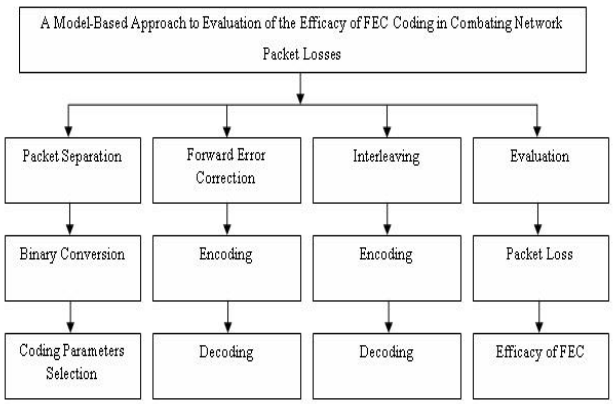

Figure 1: System architecture design

A. Concatenate FEC codes to reduce errors:

Block and convolutional codes are frequently combined in **concatenated** coding schemes in which the convolutional code does most of the work and the block code (usually Reed-Solomon) "mops up" any errors made by the convolutional decoder.

B. Turbo codes:

The most recent (early 1990s) development in error correction is turbo coding, a scheme that combines two or more relatively simple convolutional codes and an interleaver to produce a block code that can perform to within a fraction of a decibel of the Shannon limit.

- One of the earliest commercial applications of turbo coding was the CDMA2000 1x (TIA IS-2000) digital cellular technology developed by Qualcomm and sold by Verizon Wireless, Sprint, and other carriers.
- The evolution of CDMA2000 1x specifically for Internet access, 1xEV-DO (TIA IS-856), also uses turbo coding. Like 1x, EV-DO was developed by Qualcomm and is sold by Verizon Wireless, Sprint, and other carriers (Verizon's marketing name for 1xEV-DO is Broadband Access, Sprint's consumer and business marketing names for 1xEV-DO are Power Vision and Mobile Broadband, respectively.).

Interleaving: Interleaving in computer science is a way to arrange data in a non-contiguous way in order to increase performance. It is used in:

- Time-division multiplexing (TDM) in telecommunications.
- Computer memory
- disk storage

Interleaving is mainly used in data communication, multimedia file formats, radio transmission (for example in satellites) or by ADSL. The term multiplexing is sometimes used to refer to the interleaving of digital signal data.

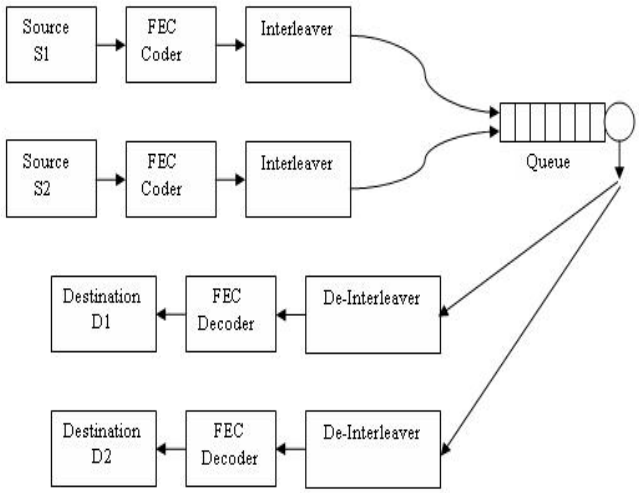

Figure2:Data Flow

Diagram Interleaving in disk storage:

Low-level format utility performing interleave speed tests on a 10-megabyte IBM PC XT hard drive.

Historically, interleaving was used in ordering block storage on disk-based storage devices such as the floppy disk and the hard disk. The primary purpose of interleaving was to adjust the timing differences between when the computer was ready to transfer data, and when that data was actually arriving at the drive head to be read. Interleaving was very common prior to the 1990s, but faded from use as processing speeds increased. Modern disk storage is not interleaved.

Interleaving was used to arrange the sectors in the most efficient manner possible, so that after reading a sector, time would be permitted for processing, and then the next sector in sequence is ready to be read just as the computer is ready to do so. Matching the sector interleave to the processing speed

therefore accelerates the data transfer, but an incorrect interleave can make the system perform markedly slower.

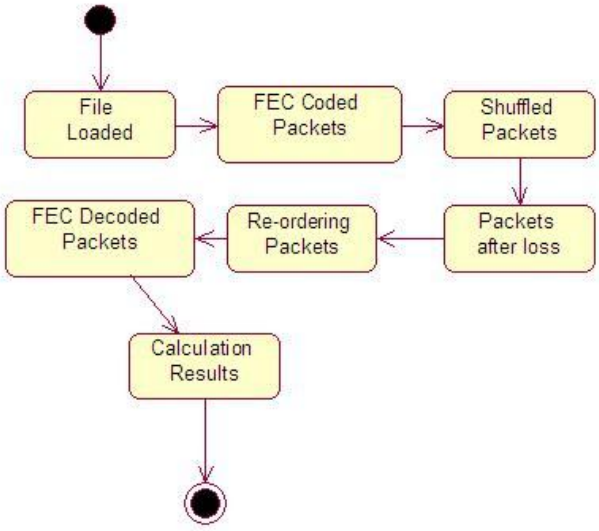

Figure 3: State Diagram

4. IMPLEMENTATION

Modules

- FEC Encoder
- Interleaver
- Implementation of the Queue
- De-Interleaver
- FEC Decoder
- Performance Evaluation

4.1 FEC Encoder:

FEC is a system of error control for data transmission, where the sender adds redundant data to its messages. This allows the receiver to detect and correct errors (within some bounds) without the need to ask the sender for additional data. In this module we add redundant data to the given input data, known as FEC Encoding.

The text available in the input text file is converted into binary. The binary conversion is done for each and every character in the input file. Then we add the redundant data for each bit of the binary. After adding we have a block of packets for each character.

The User Interface design is also done in this module. We use the Swing package available in Java to design the User Interface. Swing is a widget toolkit for Java. It is part of Sun Microsystems' Java Foundation Classes (JFC) — an API for providing a graphical user interface (GUI) for Java programs.

4.2 Interleaver:

Interleaving is a way of arranging data in a non-contiguous way in order to increase performance. It is used in data transmission to protest against burst errors. In this module we arrange the data (shuffling) to avoid burst errors which is useful to increase the performance of FEC Encoding.

This module gets the input as blocks of bits from the FEC Encoder. In this module we shuffle the bits inside a single block in order to convert burst errors into random errors. This shuffling process is done for each and every block comes from the FEC Encoder. Then we create a Socket connection to transfer the blocks from Source to the Queue. This connection is created by using the Server Socket and Socket class Available in Java.

4.3 Implementation of the Queue:

In this module we receive the data from the Source system. This data is the blocks after FEC Encoding and Interleaving processes are done. These blocks come from the Source system through Server Socket and Socket. Server socket and Socket are classes available inside Java. These two classes are used to create a connection between two systems inside a network for data transmission. After we receive the packets from Source, we create packet loss. Packet loss is a process of deleting the packets randomly. After creating loss we send the remaining blocks to the Destination through the socket connection.

4.4 De-Interleaver:

This module receives the blocks of data from the Queue through the socket connection. These blocks are the remaining packets after the loss in the Queue. In this module we re arrange the data packets inside a block in the order in which it is before Interleaving. This process of Interleaving and De-Interleaving is done to convert burst errors into random errors. After De-Interleaving the blocks are arranged in the original order. Then the data blocks are sent to the FEC Decoder.

4.5 FEC Decoder:

This module gets the input from the De-Interleaver. The received packets are processed to remove the original bits from it. Thus we recover the original bits of a character in this module. After retrieving the original bits, we convert it to characters and write it inside a text file.

4.6 Performance Evaluation:

In this module we calculate the overall performance of FEC Coding in recovering the packet losses. After retrieving the original bits, we convert it to characters and write it inside a text file. This performance is calculated by using the coding parameters like Coding rate, Interleaving depth, Block length and several other parameters. First we calculate the amount of packet loss and with it we use various formulas to calculate the overall performance of Forward Error Correction in recovering the network packet losses.

5. SIMULATION RESULT:

The tail-biting convolutional code in WiMAX systems and

binary transmission overan AWGN channel, the BP algorithm as in [4] is comparedwith the maximum-likelihood (ML) Viterbi type algorithm to decode the same tail-biting convolutional code [9, 12]. Todetermine by simulation the maximum decoding performancecapability of each algorithm, at least 300 codeword errors aredetected at each SNR. However, since the BP decoder is less complex than this traditional decoder and enables a unified decoding approach, this loss in BER performance is deemed acceptable. For further research, we propose exploring alternatives tothe flooding schedule usually adopted for LDPC codes toenhance the BER performance.

CONCLUSION

In this paper, the feasibility of decoding arbitrary tailbitingconvolutional and turbo codes using the BP algorithmwasdemonstrated. Using this algorithm to decode the tailbitingconvolutional code in WiMAX systems speeds up theerror correction convergence and reduces the decoding computationalcomplexity with respect to the ML-Viterbi-basedalgorithm. In addition, the BP algorithm performs a non-trellisbased forward-only algorithm and has only an initial decodingdelay, thus avoiding intermediate decoding delays that usuallyaccompany the traditional MAP and SOVA components inLTE turbo decoders. However, with respect to the traditionaldecoders for turbo codes, the BP algorithm is about 1.7 dBworse at a BER value of 10^{-2}. This is because the nonzeroelement distribution in the parity-check matrix is not randomenough. Also, there are a number of short cycles in thecorresponding Tanner graphs. Finally, as an extended work,we propose the BP decoder for these codes in a combinedarchitecture which is advantageous over a solution based ontwo separate decoders due to efficient reuse of computationalhardware and memory resources for both decoders. In fact,since the traditional turbo decoders (based on MAP andSOVA components) have a higher complexity, the observedloss in performance with BP is more than compensated bya drastically lower implementation complexity. Moreover, thelow decoding complexity of the BP decoder brings about endto-end efficiency since both encoding and decoding can beperformed with relatively low hardware complexity.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon limiterror-correcting coding and decoding: Turbo codes,” IEEE Intl. Conf.on Commun., vol. 2, Geneva, Switzerland, pp. 1064-1070, 1993.

[2] R. M. Tanner, “A recursive approach to low complexity codes,” IEEETrans. Inform. Theory, vol. 27, no. 5, pp. 533-547, 1981.
https://doi.org/10.1109/TIT.1981.1056404

[3] G. D. Forney, Jr., “Codes on graphs: normal realizations,” IEEE Trans.Inform. Theory, vol. 47, no. 2, pp. 520–548, 2001.
https://doi.org/10.1109/18.910573

[4] H. H. Ma and J. K. Wolf, “On Tail Biting Convolutional Codes,” IEEETrans. On Commun., vol. 34, no. 2, pp. 104-111, 1986. 3GPP TS 45.003, “3rd Generation Partnership Project; Technical SpecificationGroup GSM/EDGE Radio Access Network; Channel Coding(Release 7),” February, 2007.

[5] IEEE Std 802.16-2004, “IEEE Standard for Local andMetropolitan AreaNetworks – Part 16: Air Interface for FixedBroadband Wireless AccessSystems,” October, 2004.

[6] IEEE Std P802.16e/D10, “IEEE Standard for Local and MetropolitanArea Networks – Part 16: Air Interface for Fixed and Mobile BroadbandWireless Access Systems,” August, 2005.

[7] D. J. C. MacKay, “Good error-correcting codes based on very sparsematrices,” IEEE Trans. Inform. Theory, vol. 45, no. 2, pp. 399-431,1999.
https://doi.org/10.1109/18.748992

[8] N. Wiberg, “Codes and Decoding on General Graphs,” LinkopingStudies in Science and Technology, Dissertation No. 440, LinkopingUniversity, -Linkoping, Sweden, 1996.

[10] R. J. McEliece, D. MacKay, and J.-Fu Cheng, “Turbo Decoding as anInstance of Pearl’s "Belief Propagation" Algorithm,” IEEE Trans. OnCommun., vol. 16, no. 2, pp. 140-152, 1998.

[11] G. Colavolpe, “Design and performance of turbo Gallager codes,” IEEE Trans. On Commun., vol. 52, no. 11, pp. 1901-1908, 2004.
https://doi.org/10.1109/TCOMM.2004.836566

[12] T. T. Chen, and S-He Tsai, “Reduced-Complexity Wrap-Around Viterbi Algorithms for Decoding Tail-Biting Convolutional Codes,” 14th European Wireless Conference, Jun. 2008.

[13] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, “Improved LDPC Codes Using Irregular Graphs,” IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 585-598, 2001.
https://doi.org/10.1109/18.910576

[14] Giulietti A., “Turbo Codes: Desirable and Designable,” IKluwer Academic Publishers, ISBN: 1-4020-7660-6, 2004.
https://doi.org/10.1007/978-1-4615-0477-1

[15] P. Elias, “Coding for Noisy Channels,” IRE Conv. Rec., vol. 3, pt. 4, pp. 37–46, 1955.

[16] A. J. Viterbi, “Convolutional Codes and their Performance in Communication Systems,” IEEE Trans. On Commun., vol. 19, no. 15, pp. 751-772, 1971.
https://doi.org/10.1109/TCOM.1971.1090700

[17] C. Poulliat, D. Declercq, and T. Lestable, “Efficient Decoding of Turbo Codes with Nonbinary Belief Propagation,”EURASIPJournalonWireless Communications and Networking, vol. 2008, no. 473613, 2008.

[18] A. Refaey, S. Roy, and P. Fortier, “On the Application of BP Decoding to Convolutiona and Turbo Codes,” Asilomar Conference on Signals, Systems, and Computers, Nov. 2009.
https://doi.org/10.1109/18.796408

[19] P. Stahl, J. B. Anderson, and R. Johannesson, "Optimal and near-optimal encoders for short and moderate-length tail-biting trellises," IEEE Trans. Inform. Theory, vol. 45, no. 7, pp. 2562-2571, 1999.
[20] W. Yu, M. Ardakani, B. Smith, and F. FKschischang, "Complexityoptimized low-density parity-check codes for Gallager decoding algorithm B," IEEE International Symposium on Information Theory (ISIT)s, Sep. 2005.