FEC – Based Iterative transmission data through wireless network

B.Usha[1], S.Priyanga[2], A.Mary[3]

[*1 ,*2]*U.G Students, B.E CSE, Alpha College of Engg, Chennai, T.N, India.*

[#3]*Assistant Professor, Dept of CSE, Alpha College of Engg, Chennai, T.N, India.*

usha.ace@gmail.com

***Abstract* –** Transform data from one wireless network to another in efficient manner.Perform number of operation directly on the transformed data with the controlled loss of accuracy. Turbo codes and Forward error correcting codes (FEC) scheme for data and overhead channels. Many efficient algorithms have been proposed for decoding these codes. Introduce a universal decoder to handle these two families of codes. TCP congestion control is implemented. The present work exploits the parity-check matrix (H) representation of tail biting convolutional and turbo codes. Enabling decoding via a unified belief propagation (BP) algorithm. BP algorithm provides a highly effective general methodology for devising low-complexity iterative decoding algorithms for all convolutional code classes as well as turbo codes. Performance analysis in terms of accuracy and memory footprint.

Keywords – Traffic analysis, iterative decoding, progressive transmission.

I. INTRODUCTION

Introduced the complexity in multiple session and introduced a simple way for its implementation. The unified approach provides an integrated framework for exploring the tradeoffs between the key coding parameters: specifically, Interleaving depths, channel coding rates and block lengths. Thus by choosing the coding parameter appropriately we have achieved high performance of FEC, reduced the time delay for Encoding and Decoding with Interleaving. The error pattern or on trellis graphical representations such as in the MAP and Viterbi algorithms. The availability of network monitoring data has allowed for more complex analyses such as behavioral pattern mining, highly valuable for both content providers and communication infrastructure managers. These tasks often require to save the monitoring data; this implies storing for a long time a huge amount of information. For instance, a telecom operator willing to analyze service access patterns could easily be confronted with large data sets to be transmitted from monitoring points to processing sites, and accumulated for several months. Another example is provided by companies working in the field of Internet advertisement (ads). They have to make decisions based upon a high volume of data really fast to target ads and compute a bid price for them. The placement decision relies on patterns identified in the data logs (such as users with similar behavior as the user to present with an ad embedded in the web page he is looking at). The placement engines looks for a match if the presented ads would be clicked on with some high enough likelihood. One possible solution resorts to state-of the- art compression algorithms, in order to keep the storage footprint - and the transmission burden - manageable; but this approach has the drawback of needing a decompression stage (and then space for the uncompressed data) before any further processing. Operating on a log of a large number of users prohibit using compressed solution in real time, as the speed to present the ads is the key performance factor. In this paper we present the technique in details, including also various improvements we made recently. Moreover, we perform an in-depth performance analysis in terms of accuracy and memory footprint using different traces from different network scenarios. The results show that the transformed data closely approximates the original data while the compression ratio is close to that of the state-of-the-art compression tools.

II. RELATED WORKS

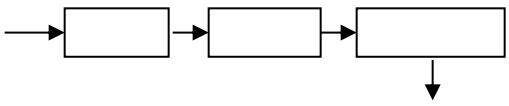
The problem of [1] is in error performance, since the last bits lack error protection. The conventional solution to this problem is to encode a fixed number of message blocks L followed by m additional all-zero blocks, where m is the constraint length of the convolutional code. The solution is in these new wireless systems, it is a good idea to introduce a universal decoder to handle these two families of codes. The problem of [2] is the scarcity of wireless bandwidth, the time-varying characteristics of the channel, and the power on wireless multimedia communications. Wireless video transmission is specially difficult because the huge volume of data required to describe a video greatly slows down transmission. The solution is bit allocation procedure that dispatches the source redundancy between the different channels when compressing to a target bit rate with a bounded side distortion. The problem of [3] is the problem of designing capacity-approaching irregular low-density parity-check (LDPC) codes under different decoding algorithms and channel models has been studied extensively. The solution is a measure is introduced that incorporates two factors that contribute to the decoding complexity. One factor, which scales linearly with the number of edges in the code's factor graph, measures the number of operations required to carry out a single decoding iteration. The problem of [4] is video streaming applications usually suffer from high packet-loss ratios due to the underlying "best-effort" model of the Internet protocol (IP). The solution is an analytical framework for evaluating FGLP bounds. Based on these bounds, shows the impact of applying fine-grained protection to the FGS enhancement-layer for different types of video sequences and over a wide range of bit-rates and packet-loss ratios.

III. DESIGN AND IMPLEMENTATION

A. FEC Encoder: FEC is a system of error control for data transmission, where the sender adds redundant data to its messages.

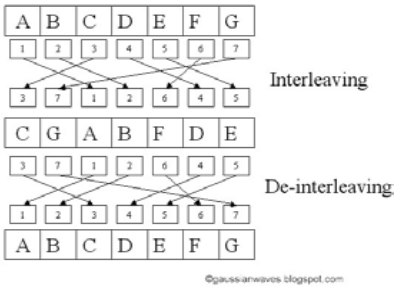
Allows the receiver to detect and correct errors (within some bounds) without the need to ask the sender for additional data. In this module we add redundant data to the given input data, known as FEC Encoding. The text available in the input text file is converted into binary. The binary conversion 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.



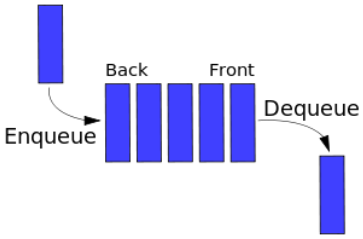
B. Interleaver:

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.



C. 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. 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.



D. 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 rearrange 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.

E. 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.

V. CONCLUSION & FUTURE WORK

With respect to the traditional decoders for turbo codes, the BP algorithm is about 1.7 dB worse at a BER value of 10−2. Because the nonzero element distribution in the parity-check matrix is not random enough. There are a number of short cycles in the corresponding Tanner graphs. we propose the BP decoder for these codes in a combined architecture which is advantageous over a solution based on two separate decoders due to efficient reuse of computational hardware and memory resources for both decoders. The traditional turbo decoders (based on MAP and SOVA components) have a higher complexity, the observed loss in performance with BP is more than compensated by a drastically lower implementation complexity.

The low decoding complexity of the BP decoder brings about end to end efficiency since both encoding and decoding can be performed with relatively low hardware complexity. Finally, as an extended work, we propose the BP decoder for these codes in a combined architecture which is advantageous over a solution based on two separate decoders due to efficient reuse of computational hardware and memory resources for both decoders. In fact, since the traditional turbo decoders (based on MAP and SOVA components) have a higher complexity, the observed loss in performance with BP is more than compensated by a drastically lower implementation complexity. Moreover, the low decoding complexity of the BP decoder brings about endto- end efficiency since both encoding and decoding can be performed with relatively low hardware complexity. With the growth of the telecommunication networks and of their user base, the need for efficiently collecting, transferring, processing and storing network monitoring data continuously poses new challenges. We presented a technique that stores high-volume network monitoring data in a representation format that satisfies two main objectives at the same time: the efficient utilization of storage space, and the possibility to perform a number of operations in a computationally convenient way and directly on the transformed data.

REFERENCES

[1]. Ahmed Refaey, Sébastien Roy, and Paul Fortier "A New Approach for FEC Decoding Based on the BP Algorithm in LTE and WiMAX Systems" Department of Electrical and Computer Engineering Université Laval, G1V 0A6, Québec, Canada

[2]. V. K. Goyal, "Multiple description coding: Compression meets the network," *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 74–93, Sep. 2001

[3]. W. Yu, M. Ardakani, B. Smith, F. FKschischang,"Complexity optimized low-density parity-check codes for Gallager decoding algorithm B".

[4]. M. van der Schaar and H. Radha, "Unequal packet loss resilience for fine-granular-scalability video," *IEEE Trans. Multimedia*, vol. 3, no. 4,pp. 381–394, Dec. 2001.

[5] D. J. C. MacKay,"Good error-correcting codes based on very sparse matrices," IEEE Trans. Inform. Theory, Vol. 45, No. 2, pp. 399-431,1999.

[6] A. J. Viterbi, "Convolutional Codes and their Performance in Communication Systems," IEEE Trans. On Commun., vol. 19, no. 15, pp. 751-772,1971.

[7] Y. Shen, P. C. Cosman, and L. B. Milstein, "Video coding with fixed length packetization for a tandem channel," *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 273–288, Feb. 2006.

[8] P. Cosman, J. Rogers, P. G. Sherwood, and K. Zeger, "Combined forward error control and packetized zerotree wavelet encoding for transmission of images over varying channels," *IEEE Trans. Image Process.*, vol. 9, no. 6, pp. 982–993, Jun. 2000.

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

[10] P. G. Sherwood and K. Zeger, "Error protection for progressive image transmission over memoryless and fading channels," *IEEE Trans. Commun.*, vol. 46, no. 12, pp. 1555–1559, Dec. 1998.