Performance Comparison of Turbo Decoding Algorithms

Alina Mirza, Shahzad Amin Sheikh
Department of Electrical Engineering, College of Electrical and Mechanical Engineering (CEME)
National University of Sciences and Technology (NUST), Pakistan
alinamirza2002@yahoo.com, sheikh.shahzadamin@gmail.com

Abstract— Turbo coding is the most commonly used error correcting scheme in wireless systems resulting in maximum coding gain. In this paper, a comparative study of the symbol-by-symbol maximum a posteriori (MAP) algorithm, its logarithmic versions, namely, Log-MAP and Max-Log-MAP decoding algorithms used in SISO Turbo Decoders are analyzed. The performance of Turbo coding algorithms are carried out in terms of bit error rate (BER) by varying parameters such as Frame size, number of iterations and choice of interleaver.

Keywords:- Iterative decoding; MAP decoding; Turbo Codes.

I. INTRODUCTION

Turbo Codes have become a popular area of development in channel coding [1] because of reaching the limits of the theoretical capacity of channel posed by C. E. Shannon in 1948. The two characteristics of the Turbo codes are concatenated scheme and iterative decoding. These codes are proposed for several applications where highly reliable transmission is required at very low signal to noise ratio (SNR). Most of the work is done to find practical decoding algorithms for implementation of these codes. The MAP algorithm is computationally complex because of numerical representation of probabilities, non-linear functions and mixed multiplications and additions of these values [3], so MAP algorithm is not a practical algorithm for implementation [4]. Some approximations of the MAP algorithms have been derived such as Log-MAP and Max-Log MAP. The logarithmic version of the MAP algorithm [3] is the practical decoding algorithms for implementation. Both of these only require addition and max-operation which can be conducted by utilizing a simple look-up table [5]. Max-Log has the least computational complexity and the worse bit error rate (BER) performance among these algorithms, while the Log-MAP algorithm [3] has the best BER performance equivalent to the MAP algorithm and the highest computational complexity.

The organization of this paper is as follows: Section II encompasses the review of the principles of turbo encoding and decoding algorithms. In section III MAP, Log-MAP and Max-Log MAP algorithms are discussed which is followed by performance comparison of Log-MAP and Max-Log MAP algorithms in section-IV.

II. TURBO ENCODING AND DECODING

A. Turbo Encoder: The code structure of Turbo code is formed by two constituent convolutional encoders concatenated in parallel through a pseudo-random interleaver. As shown in Fig.1, encoder generates parity bits from two simple recursive convolutional codes.

The information bit \( x \) is sent uncoded with two parity bits \( p_1 \) & \( p_2 \), generated from two encoders. The key innovation of turbo codes is an interleaver (I), which permutes the original bit \( x \) before input to the second encoder, due to permutation, if one encoder produces low weight code words for input sequence then other encoder will produce high weighted code words using same information sequence.

B. Turbo Decoder: After \( x, p_1, \) & \( p_2 \) are transmitted through a communication channel, they ultimately arrive at the Turbo decoder as \( y_0, y_1 \& y_2 \). Due to channel noise, the received values may differ from their transmitted values. The Turbo decoder attempts to reconstruct transmitted values through a series of decoding steps. Decoding strategy is based on the exchange of soft information between SISO (Soft Input Soft Output) component decoders in an iterative fashion. The iterative
turbo decoder structure is shown in Fig.2. Decoding is split in two steps in correspondence with the two encoding stages.

Figure 2: Iterative Turbo Decoder

The interleaver permutes the data bit to support the error correction algorithm. The output from one decoder is fed into the other decoder through an interleaver/deinterleaver to help the later decoder make a better decoding decision in subsequent decoding iterations. Multiple iterations are required before the decoder converges to a final result. After a pre-specified number of decoding iterations, the final decision is made in the hard decision block by combining the outputs from both decoders.

The component decoder takes quantized data from channel and gives out quantized confidence levels/probabilities for each decoded bit. They are therefore termed Soft Input, Soft Output (SISO) decoders.

The a-priori information is the soft inputs to the channel decoder and the a-posteriori information is the soft output of the channel decoder. The most widely used soft-values at the output of decoder are log-likelihood ratios - $L_R$ (LLR’s). They are represented as follows:

$$L_R = \ln \left( \frac{P \{ m, x_k = 1 \}}{P \{ m, x_k = 0 \}} \right)$$

(1)

If the LLR of a bit is positive, then the bit is most likely to be a ‘1’ and if it is negative, then the bit is most likely to be ‘0’.

III. TURBO DECODING ALGORITHMS

The various SISO Turbo decoders are: Maximum A-priori (MAP), Log-MAP and Max-log-MAP.

A. MAP Algorithm

The MAP algorithm provides not only the estimated sequence, but also the probabilities for each bit that has been decoded correctly. Assuming binary codes are to be used, the MAP algorithm gives, for each decoded bit $x_k$ in step k, the probability that this bit was +1 or -1, given the received distorted symbol sequence $y_0^N = (y_0, y_1, y_2, \ldots, y_N)$.

This is equivalent to finding the likelihood ratio

$$L_{x_k} = \frac{P \{ x_k = +1 \} \mid y_k^N}{P \{ x_k = -1 \} \mid y_k^N}$$

(2)

Where $P \{ x_k, \mid y_0 \} = +1, -1$ is the a posteriori probability (APP) of $x_k$.

Computation of $P \{ x_k, \mid y_0 \}$ is done by determining the probability to reach a certain encoder state $m$ after having received $k$ symbols $y_{k-1}^0 = (y_0, y_1, y_2, \ldots, y_k)$:

$$\alpha_{x_k}(m) = P \{ m \mid y_{k-1}^0 \}$$

(3)

and the probability to get from encoder state $m'$ to the final state in step $N$ with symbols $y_{k+1}$:

$$\beta_{x_k, m'}(m) = P \{ y_{k+1}^N \mid m' \}$$

(4)

The probability of the transition $m \rightarrow m'$ using the source symbol $x_k$ under knowledge of received symbol $y_k$, is called $y_k$:

$$y_k(m, m', x_k) = P \{ x_k \mid m, m', y_k \}$$

(5)

The probabilities $\alpha_k(m)$ and $\beta_k(m')$ are computed recursively over $y_k(m, m', x_k)$ which are a function of the received symbols and the channel model as below:

$$\alpha_{x_k}(m') = \sum_{m} y_k(m, m', x_k) \alpha_{x_{k-1}}(m)$$

(6)

$$\beta_{x_k, m'}(m) = \sum_{m} \beta_{x_k}(m') y_k(m, m', x_k)$$

(7)

Knowing these values for each transition $m \rightarrow m'$, the probability of having sent the symbol $x_k$ in step k the sum of all paths using the symbols $x_k$ in step k. With $\phi(x_k)$ being the set of all transitions with symbol $x_k$, we can write

$$P \{ x_k \mid y_k^N \} = \sum_{(m, m') \in \phi(x_k)} \alpha_{x_k}(m) \cdot y_k(m, m', x_k) \cdot \beta_{x_k, m'}(m')$$

(8)

Thus, we can say that MAP algorithm is complex because to evaluate the likelihood value of a decoded bit it requires many additions and multiplications.
B. Log-MAP Algorithm

The solution to the MAP algorithm problems is to operate in the log-domain. One advantage of operating in the log-domain is that multiplication becomes addition, so MAP algorithm problem of numerical instability can be avoided by operating in log-domain. The following algorithm illustrates how addition is accomplished in the log-domain.

\[ \ln (e^x + e^y) = \max(x, y) + f_s(1) \]

Where \( f_s(1) = \ln(1 + e^-1) \). So in the Log-MAP, addition is simply implemented as a maximization function plus a correction term in the log-domain. For practical implementations, this correction function can be stored in a lookup table.

Taking the negative logarithm of \( \alpha_k(m), \beta_{k+1}(m'), \gamma_k(m, m', x_k) \) and \( \lambda_k \) values from the MAP algorithm we have:

\[
\begin{align*}
A_k(m) &= -\ln \alpha_k(m) \\
B_{k+1}(m') &= -\ln \beta_{k+1}(m') \\
D_k(m, m', x_k) &= -\ln \gamma_k(m, m', x_k) \\
L_k(x_k) &= -\ln \lambda_k
\end{align*}
\]

Using the above equations, we rewrite (6) & (7) as:

\[
\begin{align*}
A_k(m) &= \sum_{m} \exp(-A_{k-1}(m) + D_k(m, m', x_k)) \\
B_{k+1}(m) &= \sum_{m', x_k} \exp(D_k(m, m', x_k) + B_{k+1}(m')) \\
D_k(m, m', x_k) &= -\ln \left[ \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[ -\frac{1}{2\sigma^2} (y_m - \hat{y}_m)^2 \right] \right] P(x_k)
\end{align*}
\]

Where \( y_m, \hat{y}_m \) are the received signal and estimated signal over a Gaussian channel respectively, \( P(x_k) \) is the a-priori probability of bit \( x_k \) and \( \sigma^2 \) is the noise variance. Using Alpha, beta and gamma log-likelihood ratio (LLR) is computed when provides soft decision, the soft output makes it possible to decide if each received bit of information is zero or one.

\[
\begin{align*}
\ln \Lambda &= \max_{(m, m') \in 01} [\ln f_{k+1}(m', m) + \ln f_{k+1}(m) + \ln \alpha_f(m', m)] \\
&\quad - \max_{(m, m') \in 01} [\ln f_{k+1}(m', m) + \ln \beta_f(m') + \ln \alpha_f(m, m)]
\end{align*}
\]

Max-Log MAP Algorithm

In the ML-MAP algorithm, the logarithm is approximated to

\[ \max^* (x, y) \sim \max(x, y) \]

As max* operation is replaced by max, i.e., remove table look-up correction term so its performance is effected. ML-MAP algorithm is simpler and faster to implement but at the expense of performance degradation.

IV. SIMULATION RESULTS

A performance comparison of the two decoding algorithms is carried out by varying the parameters (frame size, number of iterations, and choice of interleaver).

A. Frame Size
Fig. 3-4 depicts with an increase in the data frame size, improvement of decoding performance at the cost of increased latency in decoding the frame occurred [6].

**B. Number of iterations**

![Comparison of Log-Map and Max-Log Map Decoding by varying iterations](image_url)

On the other hand in Fig. 5, increase in number of iteration resulted in the improvement of decoding performance, but with an increment in computational complexity of decoding schemes [7].

**C. Effect of Choice of Interleaver**

![Comparison of Log-Map & Max-Log-Map with Block Interleaver](image_url)

![Comparison of Log-Map & Max-Log-Map Decoding Algorithms with Random interleaver](image_url)

![Comparison of Log-Map & Max-Log-Map Decoding Algorithms with Block interleaver](image_url)

![Comparison of Log-Map & Max-Log-Map Decoding Algorithms with Random interleaver](image_url)

It was shown in Fig 6-8 that random interleaver is the most appropriate interleaver to be used in Turbo encoding and decoding process. The decoding performance of random interleaver is better than diagonal and block interleaver.
V. CONCLUSION

In this paper, comparative performance evaluation of Turbo decoding Algorithms Log-MAP and Max-Log MAP is carried out. A comparison drawn between the two decoding algorithms using the defined parameters revealed that overall performance of Log-MAP decoding scheme is better than Max-Log MAP decoding scheme.

REFERENCES