The Effect Of Constraint Length And Interleaver On The Performance Of Turbo Code

Dr. Suad A. Essa *, Eman F. Khalil* & Najat Sh. Jasim*
Received on:25/9/2008
Accepted on:5/11/2009

Abstract
This paper presents a class derived from convolutional code called turbo code. The performance of turbo code is investigated through examining the effect of different constraint length, the effect of changing rate, and the effect of interleaver on the performance of turbo code with presence burst errors.
The performance of turbo code is investigates through computer simulation, by using MATLAB program.
The simulation encoder is composed of two identical RSC component encoder with parallel concatenated, separated by interleaver. The turbo code simulation results are shown graphically for different constraint length, in hard and soft decision. Also the simulation results are shown for case with interleaver and without interleaver.

*Electromechanical Engineering Department, University of Technology/Baghdad
1. Introduction
The task facing the designer of digital communication system is that of providing a cost-effective facility for transmitting information from one end of the system at a rate and a level of reliability and quality that are acceptable to the user at the other end. The two key system parameters available to the designer are the transmitted signal power and channel bandwidth, these two parameters, together with the power spectral density of noise, determine the signal energy per bit-to-noise power spectral density ratio $E_b/N_o$, this ratio determines the bit error rate, practical consideration usually place a limit on the $E_b/N_o$ value, for fixed $E_b/N_o$ the only practical option available for changing data quality from problematic to acceptable is to use error-control coding. Error control for data integrity may be exercised by means of forward error correction (FEC) [1]. (FEC) channel codes are commonly used to improve the energy efficiency of wireless communication systems. On the transmitter side, an (FEC) encoder adds redundancy to the data in the form of parity information, then at the receiver a (FEC) decoder is able to exploit the redundancy in such away that a reasonable number of channel errors can be corrected [2]. A major advancement in coding theory occurred in 1993, when a group of researchers (Berrou, Glavieux, and Thitimajshima) working in France developed turbo code. [2]

2. Channel Coding
Channel coding often used in digital communication systems to protect the digital information from noise and interference and reduce the number of bit errors. Channel coding is mostly accomplished by selectively introducing redundant bits into the transmitted information stream; these additional bits will allow detection and correction of bit errors in the received data stream and provide more reliable information transmission [3]. There are two main types of channel codes namely block codes and convolution codes.

3. Convolution Codes
Convolution codes are one of the most widely used channel codes in practical communication systems, this codes convert the entire data stream into one signal codeword, the encoded bit depend on the current $k$ input and past bits. [3] A convolution code is generated by passing the information sequence to be transmitted through a linear finite-stat shift register (flip-flop), the shift register consist of $M$ stages and $n$ linear algebraic function generators as shown in fig.(1).The input binary data to the encoder is shifted into and along the shift register $k$ bits at a time. The
number of output bits for each \( k \)-bit input sequence is \( n \) bit, the code rate is defined as \( R_c = k/n \), the parameter \( K \) is called constraint length of the convolution code, it is defined as \( K = M + 1 \) [4].

The convolution code structure is easy to draw from its parameters \((n, k, M)\), first draw \( M \) boxes representing the memory registers, then draw \( n \) modulo-2 adders to represent \( n \) output bits, then connect the memory registers to the adders using generator polynomials [5].

Let \( p(r|c) \) denote the conditional probability of receiving \( r \) given that \( c \) was sent, and \( \log p(r|c) \) is the log-likelihood function, the decision rule is described as follows:

Choose the estimate \( \hat{c} \) for which the log-likelihood function \( \log p(r|c) \) is maximum.

In the case of binary systematic channel, the transmitted code vector \( c \) and the received vector \( r \) represent binary sequences of length \( N \), and these two sequences may differ from each other in some locations because of error due to channel noise, then

\[
p(r|c) = \prod_{i=1}^{N} p(r_i|c_i) \quad [r_i, c_i]
\]

the \( i \)th element of \( r \) and \( c \).

Correspondingly the log-likelihood is

\[
\log p(r|c) = \sum_{i=1}^{N} \log p(r_i|c_i)
\]

Let the transition probability \( p(r_i|c_i) \) be defined as:

\[
p(r_i|c_i) = \begin{cases} p & \text{if } r_i \neq c_i \\ 1-p & \text{if } r_i = c_i \end{cases}
\]

If \( r \) differs from \( c \) in \( d \) positions (the number \( d \) is the hamming distance between \( r \) and \( c \)).

The log-likelihood function may rewrite as:

\[
\log p(r|c) = d \log p + (N-d) \log (1-p) \quad \log (1-p)
\]

\[
= d \log \left( \frac{p}{1-p} \right) + N \log (1-p)
\]
In general, the probability of an error occurring is low enough to assume $p < 1/2$, and $N \log(1-p)$ is constant for all $c$. Accordingly the maximum-likelihood decoding rule for the binary systematic channel as follows: choose the estimate $\hat{c}$ that minimizes the hamming distance between the received vector $r$ and the transmitted vector $c$ [1].

5. The Viterbi Algorithm:

A convolution encoder is basically a finite state machine. The optimum decoder is a maximum likelihood sequence estimator, therefore, the optimum decoding of a convolution code involves a search through the trellis for most probable sequence. Depending on whether the detector following the demodulator performs hard or soft decisions, the corresponding metric in the trellis search may be either a hamming metric or a Euclidean metric [4].

Consider the use of Viterbi algorithm for optimum decoding of the convolution encoded information sequence, for example, the trellis diagram of fig (2) for convolutional of fig (3) with rate 1/2 and constraint length 3, for each level $j$ of the trellis in range $M \leq j \leq L$ (where $M = K-1$ is the encoder’s memory, and $L$ is the length of the incoming message sequence). The survivor path and its metric for each state of the trellis are stored. The Viterbi algorithm selects the single survivor path left at the end of the process as ML path. Trace-back of the ML path on the trellis diagram would then provide decoded sequence. [3],[1]

6. Hard and Soft Decision

Hard-decision and soft decision decoding refer to the type of quantization used on the received bits, hard decision uses one-bit quantization on the received channel values, while soft decision decoding uses multi-bit quantization on the received channel values [3]. The Viterbi algorithm utilized the trellis diagram to compute the path metrics. Each state in the trellis is a signed a value. The partial path metric is determined from state $S=0$ at time $t=0$ to state $S \geq 0$. At each state, the best partial path metric is chosen from the terminated at that value. In hard-decision, Viterbi algorithm calculate the hamming distance

$$\sum_{j=1}^{n} |x_{t}^{(j)} - y_{t}^{(j)}|$$

to find $t^{th}$ branch
The Effect Of Constraint Length And Interleaver On The Performance Of Turbo Code

The encoder of a turbo code consists of two constituent systematic encoders joined together by means of an interleaver, as illustrated in fig (4). The interleaver is a critical part of the turbo code, it is a simple device that rearranges the order of the data bits. Most errors in a transmitted signal are due to noise, which usually comes in bursts. Interleaving gets rid of these bursts and spreads the errors randomly throughout the code. This is necessary since convolutional codes work very well on random or independent errors and rather poorly on bursts of errors. [6]

The interleaver can be of many types, of which the periodic and pseudo-random are two. Turbo codes use a pseudo-random interleaver, which operates only on the systematic bits. [2],[1]

The constituent codes recommended for turbo codes are short constraint length recursive systematic convolutional (RSC) codes with parallel concatenated. The reason for making the convolutional codes recursive (i.e. feeding one or more of the tap outputs in the shift register back to the input) is to make the internal state of the shift register depend on the past output. This affects the behavior of the error pattern, with the result a better performance of overall coding strategy is attained. [1]

An example of turbo-code encoder depicted in fig (5) consists of two identical RSC codes, \( C_1 \) and \( C_2 \).
which are connected to each other using parallel concatenation. Both $C_1$ and $C_2$ input use the same bit $d_k$, but due to the presence of an interleaver, input bits $d_k$ appears in different sequence. At first iteration the input sequence $d_k$ appears at both outputs $x_k$ and $y_{1k}$ or $y_{2k}$. if the encoders $C_1$ and $C_2$ are used respectively in $n_1$ and $n_2$ iteration, their rates are respectively equal to $R_1 = \frac{n_1 + n_2}{2n_1 + n_2}$, $R_2 = \frac{n_1 + n_2}{2n_2 + n_1}$.

8. Turbo code decoder

The decoder depicted in fig (6), it is made up of two elementary decoders ($DEC_1$, $DEC_2$) in serial concatenation. An interleaver installed between two decoders is used to scatter error bursts coming from $DEC_1$ output. DEMUX/INSRYION (DI) block works as a switch, redirecting input bits $y_k$ to $DEC_1$ at one moment and to $DEC_2$ at another. In off state, it feeds both $y_{1k}$ and $y_{2k}$ inputs with zeros.

For a discrete memoryless Gaussian channel and a binary modulation, the decoder receives a couple of random variables $x_k$ and $y_k$ at time $k$.

\[ x_k = (2d_k - 1) + i_k, \]
\[ y_k = (2y_k - 1) + q_k, \]

Where $i_k$ and $q_k$ are independent noise components having the same variance $\sigma^2$. $y_k$ is a $k$-th redundant information bit from $y_k$ encoder output, $y_k$ is demultiplexed and sent through (DI) to $DEC_1$ when $y_k = y_{1k}$ and to $DEC_2$ when $y_k = y_{2k}$.

The soft decoding is better than hard decoding, therefore, $DEC_1$ Yield a soft decision and delivers it to $DEC_2$. The logarithm of likelihood ratio (LLR), $\wedge (d_k)$ associated with each decoded bit $d_k$ by the first decoder $DEC_1$ is a relevant piece of information for the second decoder $DEC_2$.

\[ \wedge (d_k) = \log \frac{p_i(d_k = 1)}{p_i(d_k = 0)} \]

where $p_i(d_k = i), i = 0, 1$.

A Viterbi algorithm is unable to calculate, thus it can not be used in $DEC_1$. Instead of that, modified BCJR algorithm is used. A viterbi algorithm is an appropriate one for $DEC_2$, however the depicted structure is not optimal, because $DEC_1$ uses only a fraction of available redundant information, in order to improve the structure, a feedback loop is often used (dotted line on the figure). [7], [8]

9. Simulation Results

The simulation of the turbo code encoder is composed of two identical RSC component encoders with parallel concatenated 1/2 rate RSC code. These two component
The Effect Of Constraint Length And Interleaver On The Performance Of Turbo Code

ENCODERS are separated by random interleaver with frame size equal to the data.
The simulation results for a turbo code are based on bit error rate BER performance over a range of $E_b/N_0$.
This paper has seen examined:-
1- The influence of different constraint length ($K = 3, 4, 5, 6, 7$) on the performance of rate 1/2 RSC codes in soft and hard decision.
2- The difference between hard and soft decision influences on the performance of rate 1/2 RSC code for fixed constraint length.
3- The influence of rates 1/2 and 1/3 on the performance of RSC codes for constraint lengths ($K = 7$ and 9) in soft and hard decision.
4- The effect of interleaver on the performance of rate 1/2 RSC codes.

Table (1) show the generators polynomial and the corresponding value $d_{free}$ for several constraint lengths used to compute BER.

Fig (7) and fig (8) show the simulated performance results for the rate 1/2 RSC code in soft and hard decision respectively, for case different constraint length. Fig (9) is shown the difference in the influence between the hard and the soft decision on the performance of rate 1/2 RSC code for case fixed constraint length (7). Fig (10) and fig (11) show the simulated performance results for the rates 1/2 and 1/3 RSC code in soft and hard decision respectively. The effect of interleaver on the performance of the turbo code, is shown in fig (12) and fig (13), with introducing 7 burst errors, these figures are shown the encoded random data with interleaver and without interleaver respectively.

10. Simulation Analysis
It can be notice from figures (7) and (8) that, the performance of turbo code increases (lower in BER) with the increasing of the constraint length. From figure (9), it can be seen that the performance of turbo code in soft decision is better than in hard decision.

For case of fixed constraint length ($K = 7$) and ($K = 9$), fig. (10) and fig. (11) are shown that the performance of turbo code (in soft and hard decision) for rate 1/3 RSC code is better than the performance of turbo code for rate 1/2 RSC code.

Fig (12) and (13) are shown the response of RSC code encoder with introducing 7 burst error in random data, from comparing between these two figures, it can be seen that the interleaver gets rid of these bursts and spreads the errors randomly through the code.

11. Conclusion
1. The investigated results for the influence of the constraint length on the performance of turbo code shown that BER decreasing with increasing the constraint length, therefore the performance of the turbo code increase.
2. The investigated results in the difference between hard decision and soft decision for 7 constraint length have shown that the performance of turbo code in soft decision is better than in hard decision.
3. The investigated results for the influence of code rate in the performance of turbo code shown that BER for code rate 1/3 less than BER for code rate 1/2, therefore the performance of turbo code for rate 1/3 is better than the performance for rate 1/2. Improving of the performance of turbo code causes increasing the complexity and the cost of the turbo code, therefore due to this complexity and costing of turbo code, the performance results are for small constraint length and high code rate.

4. The investigated results of the effect of random interleaver on the turbo code performance have shown that the interleaver improves the performance of the code by reorder the data bit and spreads the error through the code.

12. References
  www.ece.cmu.edu/~dvats/M343M_Final_Report.pdf.
[8]. Turbo code, wikipedia, the free encyclopedia.
  http://en.wikipedia.org/wiki/turbo_code
The Effect Of Constraint Length And Interleaver On The Performance Of Turbo Code

Table (1) rate 1/2 maximum free distance code

<table>
<thead>
<tr>
<th>Constraint length $K$</th>
<th>Generators in octal</th>
<th>$d_{tree}$</th>
<th>Upper bound on $d_{tree}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>23</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>53</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>133</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>247</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>9</td>
<td>561</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>1,167</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>11</td>
<td>2,335</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>12</td>
<td>4,335</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>13</td>
<td>10,533</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>14</td>
<td>21,675</td>
<td>16</td>
<td>17</td>
</tr>
</tbody>
</table>

*Source: Odenwalder (1970) and Larsen (1973).*

Figure (1) : convolutional encoder
The Effect Of Constraint Length And Interleaver On The Performance Of Turbo Code

Figure (2) trellis for the convolutional encoder of figure 3

Figure (3) constraint length-3, rate-1/2 convolutional encoder
Figure (4) block diagram of turbo code

Figure (5) RSC codes encoder with Parallel concatenation
The Effect Of Constraint Length And Interleaver On The Performance Of Turbo Code

Figure (6) RSC codes decoder with serial concatenation

Figure (7) simulated performance results for 1/2 rate RSC code in soft decision viterbi decoding for different constraint length
Figure (8) simulated performance results for 1/2 rate RSC code in hard decision viterbi decoding for different constraint length.
Figure (9) simulated performance results for 1/2 rate, 7 constraint length RSC code in soft and hard decision viterbi decoding.
Figure (10) simulated performance results for 1/2 and 1/3 rates RSC code in soft decision viterbi decoding for constraint length (7, 9)
Figure (11) simulated performance results for 1/2 and 1/3 rates RSC code in hard decision viterbi decoding for constraint length (7, 9)
Figure (13) encoded data with introducing 7 burst error from 1/2 rate RSC encoder with interleaver.
Figure (13) encoded data with introducing 7 burst error from $\frac{1}{2}$ rate RSC encoder without interleaver.