

A NEW CHANNEL CODING TECHNIQUE TO APPROACH THE CHANNEL CAPACITY

Mahesh Patel¹ and A. Annamalai¹

¹Department of Electrical and Computer Engineering, Prairie View A & M University, TX 77446, United States of America

ABSTRACT

After Shannon's 1948 channel coding theorem, we have witnessed many channel coding techniques developed to achieve the Shannon limit. A wide range of channel codes is available with different complexity levels and error correction performance. Many powerful coding schemes have been deployed in the power-limited Additive White Gaussian Noise (AWGN) channel. However, it seems like we have arrived at an end of advancement path, for most of the existing channel codes. This article introduces a new coding technique that can either be used as the last coding stage of concatenated coding scheme or in parallel configuration with other powerful channel codes to achieve reliable error performance with moderately complex decoding. We will go through an example to understand the overall approach of the proposed coding technique, and finally we will look at some simulation results over an AWGN channel to demonstrate its potential.

KEYWORDS

Forward Error Correction codes, Deep space communication, Low rate channel codes, Concatenated codes.

1.INTRODUCTION

Channel codes are an indispensable part of any communication system to guarantee a reliable communication over a noisy channel. But usage of the error correcting codes requires more symbols to be transmitted, which reduces the spectral efficiency of communication system. However, the benefits have outweighed the costs and hence channel codes are deployed in most communications standards, i.e., Wireless and Mobile Communications, Deep Space Communications, and Military Communications.

The field of channel coding started with Claude Shannon's 1948 landmark paper [1]. Since then there have been many channel coding concepts and techniques evolved to achieve the channel capacity.

Concatenated coding schemes [2] were first proposed by Forney as a method for achieving large coding gains by combining two or more relatively simple constituent codes. The resulting codes had the error-correction capability of much longer codes, and they were endowed with a structure that permitted relatively easy to moderately complex decoding. The main essence of concatenated coding scheme lies in its flexible architecture that allows using different combinations of coding techniques to achieve desired performance. Turbo codes, invented by Berrou, Glavieux, and Thitimajshima [3, 4], can be imagined as a refinement of the concatenated encoding structure plus an iterative algorithm for decoding the associated code sequence. A few

years after the invention of Turbo codes, researchers became aware that Gallager's low-density parity-check (LDPC) block codes, first introduced in [5], were also capable of capacity-approaching performance on a variety of channels. LDPC codes, decoded using the belief propagation algorithm, can achieve capacity on the binary erasure channel (BEC) [6, 7] and achieve performance very close to capacity on other channels such as the binary symmetric channel (BSC) and the additive white Gaussian noise channel (AWGNC) [8]. Although Turbo codes and LDPC codes demonstrate good error correction ability, their decoding complexity is high and they require large block length to gain advantage of their decoding mechanism at low signal-to-noise ratio values. Advances in the field seem to suggest that it is very difficult to significantly improve upon existing channel coding techniques. Hence, a new class of code or perhaps a completely new approach is desirable.

This article introduces a new coding technique that has the potential to improve the error correction performance with moderate decoding complexity and approach the channel capacity.

The proposed codes can either be used as an inner code in concatenated structure or in parallel configuration with other powerful channel codes. In the proposed coding technique, we generate multiple sequences during decoding process and derive bit-wise reliability. All these sequences together with bit-wise reliability information are forwarded to the next decoding block (connected in series or parallel) to retrieve the original information bits.

A somewhat similar approach was introduced by Yasuo Harada, Qiang Wang and Vijay Bhargava [9]. They anticipated improvement in the performance of Trellis Code Modulation (TCM) serially concatenated with Reed Solomon codes [10] by introducing erasure and error decoding. They used the probability function from Hagenauer's [11] paper to declare the erasure of the symbol. For their first proposed scheme, they used the Viterbi decoder with a decision depth of $L-15$, which is 5 times the constraint length, and calculated two survivor paths with consecutive smaller minimum metrics $M1$ and $M2$. They calculated the difference between the metrics of the two paths as Δ , and when Δ is smaller than a given threshold they declared erasures at the positions where the symbols of both the paths differed. They also proposed a method of erasure declaration based on averaging the reliability information over fixed window stages. Modified Viterbi algorithms with erasure declaration were also reported in [11, 12 13 14]. However, this concept of erasure and error decoding could not work effectively at low E_b/N_o values.

This article discusses the proposed rate 1/2 codes in serially concatenated configuration. Simulation results provided in this article clearly reveal the potential of the proposed coding technique. This article exhibits the simulation results over the AWGN channel; hence we shall start with various definitions of the Shannon limit for this channel.

2. SHANNON LIMIT FOR THE AWGN CHANNEL

Performance of coding scheme over an AWGN channel may be characterized by two parameters: its signal-to-noise ratio (SNR) and its spectral efficiency (η). Here, SNR is the ratio of average signal power to average noise power, and spectral efficiency is the ratio of bit transmission rate to the channel bandwidth. If coding scheme transmits R bits per second over an AWGN channel of bandwidth W Hertz, then the spectral efficiency $\eta = R/W$ bits per second per hertz (b/s/Hz).

Shannon showed that for an AWGN channel with given SNR and bandwidth W Hertz, the rate of reliable transmission is upperbounded by

$$R < W \log_2(1 + \text{SNR})$$

Now in the power-limited regime term E_b/N_o is used more frequently, which may be defined as $E_b/N_o = \text{SNR} / \eta$; $\eta \approx 2r$ ($r = \text{coding rate}$)

For given spectral efficiency η , E_b/N_o is lower bounded by $E_b / N_o > (2^\eta - 1) / \eta$

Thus, the Shannon limit (lower bound) on E_b/N_o as a function of η is given by $(2^\eta - 1) / \eta$. This function decreases monotonically with η and approaches $\ln 2$ as $\eta \rightarrow 0$. It means the ultimate Shannon limit (lower bound) on E_b/N_o is $\ln 2$ (-1.59 dB). It is not possible to carry out reliable communication below -1.59 dB E_b/N_o value.

3. BASIC CONFIGURATION FOR THE PROPOSED CHANNEL CODES

Figure 1 illustrates the basic block diagram of serially concatenated configuration with the proposed codes as an inner code. Input bits go through two coding blocks during the process: 1) outer code and 2) the proposed inner code.

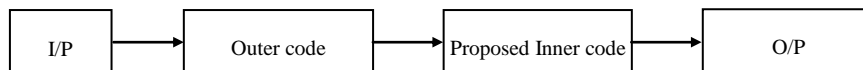


Figure 1. The proposed codes in serially concatenated configuration

Figure 2 illustrates the basic block diagram of parallel concatenated configuration using the proposed codes. Any existing channel code or combination of channel codes can be combined with the proposed codes, either in parallel configuration or serial configuration. However, the decoding logic of the selected coding technique, in both the cases, needs to be modified in order to make full use of multiple sequences and bit-wise reliability information provided by the proposed code. It will become more evident once we go through the example. In order to evaluate the proposed coding technique, the performance of bit-wise reliability sequences is more important than its relative position in the coding structure. One can also use an interleaver in conjunction with the existing channel code/ outer code block.

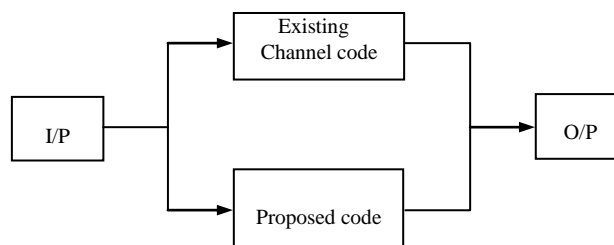


Figure 2. The proposed codes in parallel concatenated configuration

This article only discusses the serially concatenated configuration to introduce the proposed coding technique; hence the proposed codes are referred to as proposed inner code for the rest of the article. At the transmitter side, information bits are encoded using the serially concatenated

configuration . During the encoding process, the information bits go through outer encoder and inner encoder blocks respectively.

3.1 Outer Encoder

Code rate of the outer code depends on the selection of the outer coding technique. We can either use a single channel coding technique or a combination of multiple channel codes as an outer code. Any powerful channel codes like LDPC codes, Turbo codes, Raptor codes [15] or combination of other channel codes can be used as an outer code. This article does not emphasise on end-to-end system realization, hence the selection criterion for the outer code is not discussed in detail.

3.2 Inner Encoder

Code rate of the proposed inner code is 1/2 and its trellis representation is illustrated in Figure 3. There are four states and each input bit is encoded using two encoding bits dictated by its ending state. One can also alter the bit notations for the states. Encoding process always starts from state ‘S0’. A solid line in the trellis represents bit ‘1’ while a dotted line represents bit ‘0’. From the trellis illustrated in Figure 3, it is clear that the proposed rate 1/2 inner code is a systematic code.

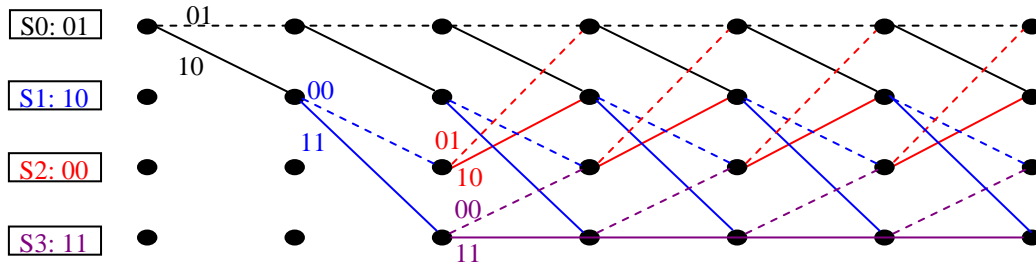


Figure 3. Rate 1/2 inner code trellis structure

Although the trellis representation of the proposed inner encoding technique may look quite similar to rate 1/2 convolutional code [16], the actual encoding process is completely different. The traditional convolutional codes are mostly decoded using Viterbi algorithm (VA), invented by Andy Viterbi in 1977 [17]. However, the proposed inner code uses a unique decoding approach of generating multiple sequences and deriving bit-wise reliability. An example to understand the proposed inner encoding and decoding process in detail is provided below.

Assume that after performing outer encoding we got the following bit sequence as an input to the proposed inner encoder:

10011001100010110001111001000010101010101001110001000000100111011010 [70 bits].

Performing the inner encoding on this bit sequence:

e.g. The starting state is always S0,

The first input bit is 1 and the present state is S0. Therefore, the encoded bits are ‘10’ and the next state becomes S1

The second input bit is 0 and the present state is S1. Therefore, the encoded bits are '00' and the next state becomes S2

The third input bit is 0 and the present state is S2. Therefore, the encoded bits are '01' and the next state becomes 'S1'

The fourth input bit is 1 and the present state is S1. Therefore, the encoded bits are '10' and the next state becomes 'S2'....

Finally we get the following inner encoded sequence:

10 00 01 10 11 00 01 10 11 00 01 01 10 00 10 11 00 01 01 10 11 11 11 00 01 10 00 01 01 01 10
 00 10 00 10 00 10 00 10 00 10 00 01 10 11 11 00 01 01 10 00 01 01 01 01 01 10 00 01 10
 11 11 00 10 11 00 10 00 [70*2=140 bits]

These encoded bits are modulated and transmitted to the receiver through the AWGN channel.

4. PROPOSED DECODING TECHNIQUE

When message bits are transmitted through a noisy channel, some symbols are corrupted by the noise and hence we get few bits in error at the receiver end. For our example, suppose we get the following bit sequence at the receiver after performing demodulation and hard decision on the received signal: 10 00 01 **11** 11 00 01 10 **01** 00 **11** 01 10 00 10 **10** 00 01 01 10 11 11 11 **11** 01 10
 00 01 **00** 01 10 00 10 00 **01** 00 10 00 10 00 **11** 00 10 00 01 **11** 11 11 00 01 01 10 **11** **00** 01 01 01
00 **11** 00 01 10 11 11 00 10 11 00 10 00 [total 140bits: Bits marked in bold are in error].

While decoding serially concatenated codes, the inner code protects the symbols of the outer code and reduces the error events, so that outer code can easily correct rest of the errors and retrieve the original information bits. As mentioned earlier this article does not discuss much about the outer decoding as we can select the existing coding technique as an outer code.

Basic block diagram of the proposed serially concatenated decoding process is illustrated in Figure 4. This article focuses only on hard decision based inner decoding. Demodulator block provides the hard-decision bits to the proposed inner decoder block.

During the proposed inner decoding process, multiple sequences are decoded using hard-decision bits. Different combinations of these decoded sequences are used to generate the bit-wise reliability sequences.

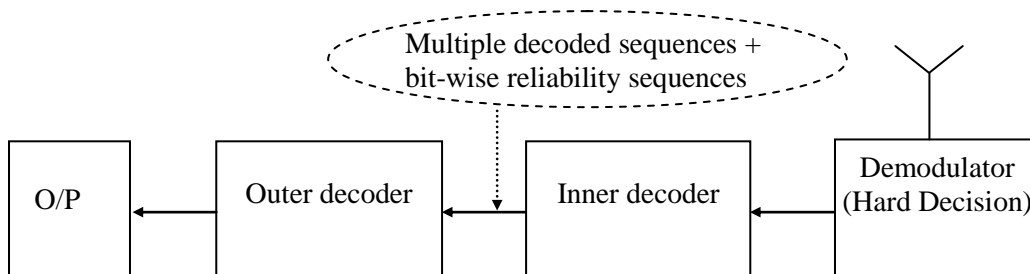


Figure 4. Basic block diagram of the proposed serially concatenated decoding process

4.1 Proposed inner decoding technique and the simulation results

Flowchart of the proposed inner decoding process for N received bits, where N can be any even integer, is shown in Figure 5. It is obvious that performing inner decoding on N received bits will give $N/2$ decoded bits as a decoded output.

As shown in the flowchart, during the inner decoding process five sequences, forward path-1 (FP-1), forward path-2 (FP-2), forward path-3 (FP-3), reverse path-1 (RP-1), and reverse path-2 (RP-2), are generated using different logical functions. Using collective information from selected decoded sequences, two bit-wise reliability sequences called accurate-bits sequence (ABS) and highly accurate-bits sequence (HABS) are generated. We can derive additional bit-wise reliability sequences using different combinations of the decoded sequences.

We will now go through the logical functions and procedures for decoding multiple sequences, and generating bit-wise reliability sequences. As the inner encoding rate is $\frac{1}{2}$, the decoding process is performed using sets of two bits. After receiving the bits string from the demodulator block, the first step in the proposed inner decoding process is to create the sets that contain two bits each. From encoding trellis, we can proclaim the valid transitions and invalid transitions between any two successive sets or states as depicted in Table 1, where n represents an odd integer $0 < n < N-2$.

Table 1. Acceptable and unacceptable transition conditions

Bit No. 'n, n+1'	Bit No. 'n+2, n+3'	Validation of transition between two consecutive sets
0 1	0 0	Invalid
0 1	0 1	Valid
0 1	1 0	Valid
0 1	1 1	Invalid
1 0	0 0	Valid
1 0	0 1	Invalid
1 0	1 0	Invalid
1 0	1 1	Valid
1 1	0 0	Valid
1 1	0 1	Invalid
1 1	1 0	Invalid
1 1	1 1	Valid
0 0	0 0	Invalid
0 0	0 1	Valid
0 0	1 0	Valid
0 0	1 1	Invalid

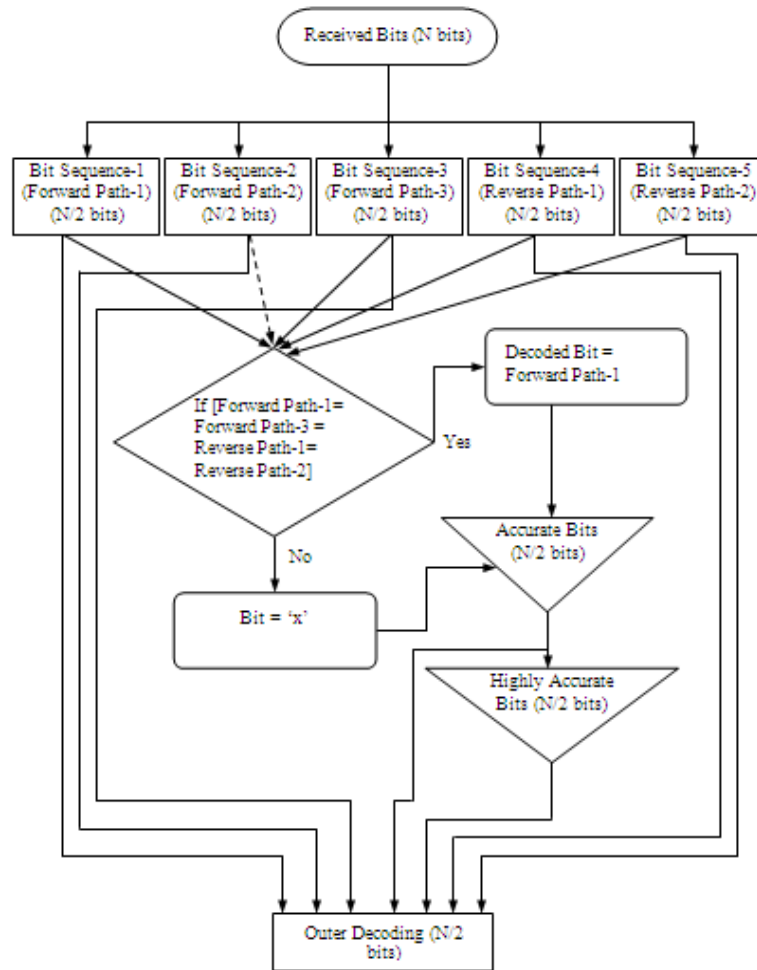


Figure 5. Flowchart of the proposed inner decoding process

After generating all the sets, the next step is to fix and decode the first and last set.

4.1.1. Decoding the first set

From the inner encoding trellis, it is obvious that the first two bits can be either 10 or 01. If the first set (the first two bits) is 11 or 00, then modify it to 01 or 10 depending on the bits at position 3 and position 4. Table 2 explains the complete decoding logic for the first set.

Table 2. Decoding logic for the first set

The First Set: Bit No. '1, 2'	Bit No. '3, 4'	Modify Bit No. '1, 2'	Decoded Bit No. '1'
0 1	Don't care	0 1	0
1 0	Don't care	1 0	1
1 1	1 0	0 1	0
1 1	0 1	0 1	0
1 1	1 1	1 0	1
1 1	0 0	1 0	1
0 0	1 0	0 1	0
0 0	0 1	0 1	0
0 0	1 1	1 0	1
0 0	0 0	1 0	1

4.1.2. Decoding the last set

Keep the last set (the last two bits) as it is and decode them as shown in Table 3.

Table 3. Decoding logic for the last set

The Last Set: Bit No. 'N-1, N'	Decoded Bit No. 'N/2'
0 1	0
1 0	1
1 1	1
0 0	0

4.1.3. Decoding multiple sequences using the rest of the sets

After decoding the first and last set, different logical operations are performed on the rest of the sets (demodulated bits: '3, 4 ... N-3, N-2') to decode multiple sequences. We use the same demodulated bits or sets to generate all five sequences. However, while generating different sequences, the original bits may be modified depending on the logical function of that particular sequence.

During the inner decoding, we represent the decoded bit using one of the three statuses: bit '0', bit '1' or discarded bit 'x'. If the bit is decoded as '1' or '0', it is considered as a valid bit or reliable bit, and if it is decoded as 'x', then that particular bit position is considered as a discarded bit or unreliable bit.

4.1.4. Forward path-1 (FP-1)

While generating this sequence, we proceed from left to right, in a forward direction, so it is called forward path-1. We decode a particular set based on its transition condition with the preceding set. While proceeding in forward direction, if a valid transition occurs between two successive sets, then decode the set under operation using Table 3. If an invalid transition occurs, then decode the set under operation using the sets before and after that location. Modify the bits of the set under operation during the decoding process as suggested in Table 4. A complete logical algorithm for decoding forward path-1 is shown in Table 4, where n is an odd integer $2 < n < N-2$ (i.e. second set represents demodulated bit number 3, and 4; $n=3$).

e.g. Assume while decoding set 3 (demodulated bits ‘5, 6’; n=5), invalid transition occurred between set 2 and set 3. Therefore, we shall decode and modify set 3 based on the bits of set 2 (demodulated bits ‘4, 5’) and set 4 (demodulated bits ‘7, 8’).

Table 4. Decoding algorithm for forward path-1

Bit No. ‘n-2, n-1’	Bits Under Operation ‘n, n+1’	Bit No. ‘n+2, n+3’	Modify Bit No. ‘n, n+1’	Decoded Sequence ‘Forward Path-1’ Bit No. ‘[(n-1)/2] + 1’
0 1	0 1	0 1	0 1	0
0 1	0 1	1 0	0 1	0
0 1	0 1	1 1	0 1	0
0 1	0 1	0 0	0 1	0
0 1	1 0	0 1	1 0	1
0 1	1 0	1 0	1 0	1
0 1	1 0	1 1	1 0	1
0 1	1 0	0 0	1 0	1
0 1	1 1	0 1	0 1	0
0 1	1 1	1 0	0 1	0
0 1	1 1	1 1	1 0	1
0 1	1 1	0 0	1 0	1
0 1	0 0	0 1	0 1	0
0 1	0 0	1 0	0 1	0
0 1	0 0	1 1	1 0	1
0 1	0 0	0 0	1 0	1
1 0	0 1	0 1	0 0	0
1 0	0 1	1 0	0 0	0
1 0	0 1	1 1	1 1	1
1 0	0 1	0 0	1 1	1
1 0	1 0	0 1	0 0	0
1 0	1 0	1 0	0 0	0
1 0	1 0	1 1	1 1	1
1 0	1 0	0 0	1 1	1
1 0	1 1	0 1	1 1	1
1 0	1 1	1 0	1 1	1
1 0	1 1	1 1	1 1	1
1 0	1 1	0 0	1 1	1
1 0	0 0	0 1	0 0	0
1 0	0 0	1 0	0 0	0
1 0	0 0	1 1	0 0	0
1 0	0 0	0 0	0 0	0
1 1	0 1	0 1	0 0	0
1 1	0 1	1 0	0 0	0
1 1	0 1	1 1	1 1	1
1 1	0 1	0 0	1 1	1
1 1	1 0	0 1	0 0	0
1 1	1 0	1 0	0 0	0
1 1	1 0	1 1	1 1	1
1 1	1 0	0 0	1 1	1
1 1	1 1	0 1	1 1	1
1 1	1 1	1 0	1 1	1
1 1	1 1	1 1	1 1	1
1 1	1 1	0 0	1 1	1
1 1	0 0	0 1	0 0	0

Bit No. 'n-2, n-1'	Bits Under Operation 'n, n+1'	Bit No. 'n+2, n+3'	Modify Bit No. 'n, n+1'	Decoded Sequence 'Forward Path-1' Bit No. '[(n-1)/2] + 1'
1 1	0 0	1 0	0 0	0
1 1	0 0	1 1	0 0	0
1 1	0 0	0 0	0 0	0
0 0	0 1	0 1	0 1	0
0 0	0 1	1 0	0 1	0
0 0	0 1	1 1	0 1	0
0 0	0 1	0 0	0 1	0
0 0	1 0	0 1	1 0	1
0 0	1 0	1 0	1 0	1
0 0	1 0	1 1	1 0	1
0 0	1 0	0 0	1 0	1
0 0	1 1	0 1	0 1	0
0 0	1 1	1 0	0 1	0
0 0	1 1	1 1	1 0	1
0 0	1 1	0 0	1 0	1
0 0	0 0	0 1	0 1	0
0 0	0 0	1 0	0 1	0
0 0	0 0	1 1	1 0	1
0 0	0 0	0 0	1 0	1

4.1.5. Forward path-2 (FP-2)

While generating this sequence, we again proceed from left to right, in a forward direction, so it is called forward path-2. We decode and modify a set under operation using the sets before and after that location, without considering its transition condition. A complete logical algorithm for decoding forward path-2 is shown in Table 5, where n is an odd integer $2 < n < N-2$.

e.g. For set 3 ($n=5$), decode and modify the bits of set 3 based on the bits of set 2 and set 4.

Table 5. Decoding algorithm for forward path-2

Bit No. 'n-2, n-1'	Bits Under Operation 'n, n+1'	Bit No. 'n+2, n+3'	Modify Bit No. 'n, n+1'	Decoded Sequence 'Forward Path-2' Bit No. '[(n-1)/2] + 1'
0 1	Don't care	0 1	0 1	0
0 1	Don't care	1 0	0 1	0
0 1	Don't care	1 1	1 0	1
0 1	Don't care	0 0	1 0	1
1 0	Don't care	0 1	0 0	0
1 0	Don't care	1 0	0 0	0
1 0	Don't care	1 1	1 1	1
1 0	Don't care	0 0	1 1	1
1 1	Don't care	0 1	0 0	0
1 1	Don't care	1 0	0 0	0
1 1	Don't care	1 1	1 1	1
1 1	Don't care	0 0	1 1	1
0 0	Don't care	0 1	0 1	0
0 0	Don't care	1 0	0 1	0
0 0	Don't care	1 1	1 0	1
0 0	Don't care	0 0	1 0	1

4.1.6. Forward path-3 (FP-3)

While generating this sequence, we continue to proceed from left to right, in a forward direction, so it is called forward path-3. We decode a set under operation based on its transition condition with the preceding set. While proceeding in forward direction, if a valid transition occurs between the set under operation and its preceding set, then decode that particular set using Table 3. If an invalid transition occurs, then decode the set under operation with 'x' (discarded-bit). A complete logical algorithm for decoding forward path-3 is shown in Table 6, where n is an odd integer $2 < n < N-2$.

e.g. Assume while decoding set number 3 ($n=5$), invalid transition occurred between set 2 and set 3. Therefore, we shall decode set 3 with 'x'.

Table 6. Decoding algorithm for forward path-3

Bit No. 'n-2, n-1'	Bits Under operation 'n, n+1'	Decoded Sequence 'Forward Path-3' Bit No. ' $[(n-1)/2] + 1$ '
0 1	0 0	x
0 1	0 1	0
0 1	1 0	1
0 1	1 1	x
1 0	0 0	0
1 0	0 1	x
1 0	1 0	x
1 0	1 1	1
1 1	0 0	0
1 1	0 1	x
1 1	1 0	x
1 1	1 1	1
0 0	0 0	x
0 0	0 1	0
0 0	1 0	1
0 0	1 1	x

4.1.7. Reverse path-1 (RP-1)

While generating this sequence, we proceed from right to left, in a reverse direction, so it is called reverse path-1. The overall logic for decoding reverse path-1 is similar to that of forward path-1, except it is performed in the reverse direction. A complete logical algorithm for decoding reverse path-1 is shown in Table 7, where n is an odd integer with descending order $N-2 > n > 2$.

e.g. Assume while decoding set 3, invalid transition occurred between set 2 and set 3. So we shall decode and modify set 3 based on bits of set 4 and set 2.

Table 7. Decoding algorithm for reverse path-1

Bit No. 'n+2, n+3'	Bits Under Operation 'n, n+1'	Bit No. 'n-2, n-1'	Modify Bit No. 'n, n+1'	Decoded Sequence 'Reverse Path-1' Bit No. ' $[(n-1)/2] + 1$ '
0 1	0 1	0 1	0 1	0
0 1	0 1	1 0	0 1	0
0 1	0 1	1 1	0 1	0
0 1	0 1	0 0	0 1	0
0 1	1 0	0 1	0 1	0
0 1	1 0	1 0	0 0	1
0 1	1 0	1 1	0 0	1
0 1	1 0	0 0	0 1	1
0 1	1 1	0 1	0 1	0
0 1	1 1	1 0	0 0	0
0 1	1 1	1 1	0 0	1
0 1	1 1	0 0	0 1	1
0 1	0 0	0 1	0 0	0
0 1	0 0	1 0	0 0	0
0 1	0 0	1 1	0 0	1
0 1	0 0	0 0	0 0	1
1 0	0 1	0 1	0 1	0
1 0	0 1	1 0	0 1	0
1 0	0 1	1 1	0 1	1
1 0	0 1	0 0	0 1	1
1 0	1 0	0 1	0 1	0
1 0	1 0	1 0	0 0	0
1 0	1 0	1 1	0 0	1
1 0	1 0	0 0	0 1	1
1 0	1 1	0 1	0 1	1
1 0	1 1	1 0	0 0	1
1 0	1 1	1 1	0 0	1
1 0	1 1	0 0	0 1	1
1 0	0 0	0 1	0 0	0
1 0	0 0	1 0	0 0	0
1 0	0 0	1 1	0 0	0
1 0	0 0	0 0	0 0	0
1 1	0 1	0 1	1 0	0
1 1	0 1	1 0	1 1	0
1 1	0 1	1 1	1 1	1
1 1	0 1	0 0	1 0	1
1 1	1 0	0 1	1 0	0
1 1	1 0	1 0	1 0	0
1 1	1 0	1 1	1 0	1
1 1	1 0	0 0	1 0	1
1 1	1 1	0 1	1 1	1
1 1	1 1	1 0	1 1	1
1 1	1 1	1 1	1 1	1
1 1	1 1	0 0	1 1	1
1 1	0 0	0 1	1 0	0
1 1	0 0	1 0	1 1	0
1 1	0 0	1 1	1 1	0
1 1	0 0	0 0	1 0	0
0 0	0 1	0 1	1 0	0

Bit No. 'n+2, n+3'	Bits Under Operation 'n, n+1'	Bit No. 'n-2, n-1'	Modify Bit No. 'n, n+1'	Decoded Sequence 'Reverse Path-1' Bit No. ' $[(n-1)/2] + 1$ '
0 0	0 1	1 0	1 1	0
0 0	0 1	1 1	1 1	0
0 0	0 1	0 0	1 0	0
0 0	1 0	0 1	1 0	1
0 0	1 0	1 0	1 0	1
0 0	1 0	1 1	1 0	1
0 0	1 0	0 0	1 0	1
0 0	1 1	0 1	1 1	0
0 0	1 1	1 0	1 1	0
0 0	1 1	1 1	1 1	1
0 0	1 1	0 0	1 1	1
0 0	0 0	0 1	1 0	0
0 0	0 0	1 0	1 1	0
0 0	0 0	1 1	1 1	1
0 0	0 0	0 0	1 0	1

4.1.8. Reverse path-2 (RP-2)

While generating this sequence, we again proceed from right to left, in a reverse direction, so it is called reverse path-2. The overall logic for decoding reverse path-2 is similar to that of forward path-2, except it is performed in the reverse direction. A complete logical algorithm for decoding reverse path-2 is shown in Table 8, where n is an odd integer with descending order $N-2 > n > 2$. e.g. For set 3 ($n=5$), decode and modify the bits of set 3 based on the bits of set 4 and set 2.

Table 8. Decoding algorithm for reverse path-2

Bit No. 'n+2, n+3'	Bits Under Operation 'n, n+1'	Bit No. 'n-2, n-1'	Modify Bit No. 'n, n+1'	Decoded Sequence 'Reverse Path-2' Bit No. ' $[(n-1)/2] + 1$ '
0 1	Don't care	0 1	0 1	0
0 1	Don't care	1 0	0 0	0
0 1	Don't care	1 1	0 0	0
0 1	Don't care	0 0	0 1	0
1 0	Don't care	0 1	0 1	0
1 0	Don't care	1 0	0 0	0
1 0	Don't care	1 1	0 0	0
1 0	Don't care	0 0	0 1	0
1 1	Don't care	0 1	1 0	1
1 1	Don't care	1 0	1 1	1
1 1	Don't care	1 1	1 1	1
1 1	Don't care	0 0	1 0	1
0 0	Don't care	0 1	1 0	1
0 0	Don't care	1 0	1 1	1
0 0	Don't care	1 1	1 1	1
0 0	Don't care	0 0	1 0	1

For the received bits in our example, the five decoded sequences are:

FP-1:

10111000101010010101111001011010101010101011110001100000100111011010

FP-2:

1001100011001011010111110100001010001011101011110001100000100111011010

FP-3:

100x1001xxxx101x00011111x100x010100x1010x0100x11000110000xx00111011010

RP-1:

1011100110101001010111100101001010101011101001110001100011100111011010

RP-2:

1001100101001011010111100101001010001010101001110001100010100111011010

The bit-wise reliability sequences (ABS and HABS) are generated using different combinations of multiple decoded sequences. A bit in the accurate-bits sequence (ABS) represents bit '0' if all the selected paths represent bit '0', or bit '1' if all the selected paths represent bit '1'. If any path out of the selected paths has a different bit, then that bit position is represented by 'x'. The Valid bits (bits presented by either '0' or '1') in the ABS have less error probability compared to any individual decoded sequence; hence the valid bits should be treated with more significance than the discarded-bits during the outer decoding. For the given example, we get the following ABS.

10xx100xxxxx10xx0001111xx10xx01010xx101xx010xx1100011000xx100111011010

We will now compare the ABS with the original encoded sequence.

100110011000101100011110010000101010101010100111000100000100111011010

10xx100xxxxx10xx0001111xx10xx01010xx101xx010xx1100011000xx100111011010

There is only one bit in error out of 49 ($[49/70]*100 = 70\%$) valid bits, so BER of the ABS for given example is $1/49 = 0.02041$.

HABS is generated by discarding the valid bits that are immediately before or after the discarded bit positions in the ABS. The HABS for a given example is:

1xxxx0xxxxxxxxxxxx00111xxxxxxxx101xxxx0xxxx1xxxx10001100xxxx00111011010

In order to understand the significance of deriving bit-wise reliability, let us examine the simulation results of the ABS and HABS over the AWGN channel. It is evident that the use of coding (redundancy bits) decreases the signal-to-noise ratio that is used to transmit over the channel. As the total coding rate depends on the selection of outer code, which is not implemented in this article, it is not possible to add appropriate noise level to the transmitted signal. Hence, we should consider performance (BER) improvement, achieved by the ABS and HABS, with respect to demodulated bit error rate rather than its apparent E_b/N_o value. Table 9 shows the simulation results of the accurate- bits sequence (ABS) and highly accurate-bits sequence (HABS), generated using FP-1, FP-3, RP-1, and RP-2. Frame size of 1200 bits, BPSK modulation, and 1000 iterations were used during all the simulations.

Table 9. Simulation results of the ABS and HABS generated using FP-1, FP-3, RP -1 and RP-2

E_b/N_o (dB)	Demodulated BER	ABS		HABS	
		BER	Valid bits %	BER	Valid bits %
-7	0.2636	0.1447	39.36	--	< 40
-6.5	0.2516	0.1297	40.66	--	< 40
-6	0.2397	0.1112	41.70	--	< 40
-5.5	0.2266	0.0948	43.09	--	< 40
-5	0.2134	0.0794	44.82	--	< 40
-4.5	0.2001	0.0651	46.40	--	< 40
-4	0.1866	0.0526	48.50	--	< 40
-3.5	0.1726	0.0418	50.91	--	< 40
-3	0.1590	0.0320	53.24	--	< 40
-2.5	0.1442	0.0244	56.13	--	< 40
-2	0.1304	0.0182	59.21	--	< 40
-1.5	0.1170	0.0124	62.33	--	< 40
-1	0.104	0.0087	65.42	0.0028	42.22
-0.5	0.0915	0.0061	68.26	0.0018	46.14
0	0.0791	0.0040	71.78	0.0012	51.24
0.5	0.0668	0.0022	76.01	0.0006284	57.61
1	0.0563	0.0013	79.25	0.0003472	62.70
1.5	0.0466	0.000734	82.65	0.0002479	68.28

Table 10 shows the simulation results of the ABS and HABS generated using FP-1, FP-2, RP-1 and RP-2. For both the cases, we only have presented the HABS BER performance results that have the amount of valid bits greater than 40 percent.

Table 10. Simulation results of the ABS and HABS generated using FP-1, FP-2, RP-1 and RP-2

E_b/N_o (dB)	Demodulated BER	ABS		HABS	
		BER	Valid bits %	BER	Valid bits %
-7	0.2636	0.1846	55.35	--	< 40
-6.5	0.2516	0.1687	56.34	--	< 40
-6	0.2397	0.1539	57.18	--	< 40
-5.5	0.2266	0.1373	58.12	--	< 40
-5	0.2134	0.1220	59.62	--	< 40
-4.5	0.2001	0.1058	60.92	--	< 40
-4	0.1866	0.0918	62.46	--	< 40
-3.5	0.1726	0.0779	64.11	--	< 40
-3	0.1590	0.0654	65.91	--	< 40
-2.5	0.1442	0.0531	68.14	--	< 40
-2	0.1304	0.0421	70.24	--	< 40
-1.5	0.1170	0.0348	72.17	--	< 40
-1	0.104	0.0263	74.84	--	< 40
-0.5	0.0915	0.0196	77.31	0.0027	40.83
0	0.0791	0.0146	79.93	0.0017	45.00
0.5	0.0668	0.01	82.57	0.00098	51.33
1	0.0563	0.0072	84.98	0.00055	57.00
1.5	0.0466	0.005	87.37	0.00036	60.17

Figure 5 shows a more conventional representation of the ABS performance, obtained in Tables 9 and 10.

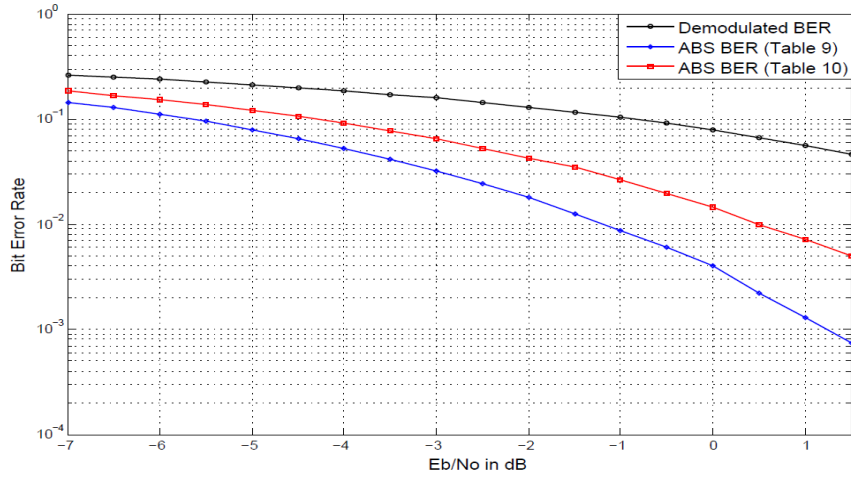


Figure 5. Performance comparison: Demodulated BER Vs ABS BER.

As can be seen from Figure 5, the ABS brings down the error rate even for higher values of demodulated BER (demodulated BER > 0.13). The ABS generated using FP-1, FP-3, RP-1 and RP-2 (Table 9) provide better BER performance than that of generated using FP-1, FP-2, RP-1 and RP-2 (Table 10), but with less amount of valid bits. We can use multiple bit-wise reliability sequences for the outer decoding but that will make the outer decoding process a bit complex, hence only one ABS is advisable. The proposed rate 1/2 inner code performs consistently, in terms of the ABS BER, even for higher values of demodulated BER, i.e., demodulated BER > 0.13. This is an interesting observation as the Shannon-limit for rate 1/2 code is 0.2 dB. It signifies the lowest possible bit-energy-to-noise ratio E_b/N_o , required to achieve a reliable BER over the binary-input AWGN channel using rate 1/2 codes.

The concept of bit-wise reliability may look similar to erasure declaration logic. However, in the erasure declaration method we do not consider the erased bits for the decoding process. While in the proposed technique, even though we have discarded few bit positions in the ABS, the other decoded sequences (FP-1&2 and RP-1&2) still have a valid bit for all the locations.

The ABS does not have a valid bit (either '0' or '1') for all the bit positions; therefore, the proposed inner code cannot be used alone. There must be some encoding stage combined with it, which can offer high code distance. In a serially concatenated coding configuration the bit-wise reliability information (ABS and HABS) and other decoded sequences (FP-1, FP-2, FP-3, RP-1, and RP-2) are forwarded to the outer decoding block.

It should be emphasized that although we focus our performance analysis on BPSK transmission over various channels, the techniques we consider here can be implemented in a straightforward manner using any modulation such as multiple phase shift keying (MPSK), quadrature amplitude modulation (QAM) or frequency shift keying (FSK).

4.2 Outer Decoding

Once we have the bit-wise reliability sequences and other decoded sequences, outer decoding should become easy. As discussed earlier any high distance code or error erasure code can be used as outer code. We can decode the outer coding stage using collective information from bit-wise reliability sequences and multiple decoded sequences.

Outer encoded sequence, used in our example, should be separated from other possible legal codes by finite code distance. To show the effectiveness of the proposed inner coding scheme we shall compute the hamming distance between different decoded sequences and original encoded bit sequence. However, in practice we will have multiple legitimate sequences/patterns to choose from, based on the size of the outer decoding window. This will make the decoding process more complex than what we will discuss next.

Original outer encoded bits:

1001100110001011000111100100001010101010101001110001000000100111011010

Assuming 10 bits window size for outer decoding, the original outer encoded window patterns are:

- Window 1: 1001100110
- Window 2: 0010110001
- Window 3: 1110010000
- Window 4: 1010101010
- Window 5: 1010011100
- Window 6: 0100000010
- Window 7: 0111011010

We will perform the decoding operation for the last two windows, window 6 and 7, as there is an error event within the ABS for window 6. Here we have considered FP-1, FP-2, RP-1, and RP-2 for the outer decoding. For window 6, outer encoded pattern is ‘0100000010’ and its Hamming distance with respect to different decoded sequences is listed in Table 11.

Table 11. Code distance of different decoded sequences with respect to Window 6

Decoded Sequence	Comparison	Code Distance in Bits
Forward Path-1	0110000010 0100000010 (W6)	1
Forward Path-2	0110000010 0100000010 (W6)	1
Reverse Path-1	0110001110 0100000010 (W6)	3
Reverse Path-2	0110001010 0100000010 (W6)	2
Accurate-bits sequence	011000xx10 0100000010 (W6)	1

If we assume that the outer code provides sufficiently large code distance, then it is obvious from the comparison, shown in Table 11, that FP-1 or FP-2 together with the ABS strongly suggest the selection of given outer encoded pattern.

For window 7, the outer encoded pattern is '0111011010' and its Hamming distance with respect to different decoded sequences is listed in Table 12.

Table 12. Code distance of different decoded sequences with respect to Window 7

Decoded Sequence	Comparison	Code Distance in Bits
Forward Path-1	0111011010 0111011010 (W7)	0
Forward Path-2	0111011010 0111011010 (W7)	0
Reverse Path-1	0111011010 0111011010 (W7)	0
Reverse Path-2	0111011010 0111011010 (W7)	0
Accurate-bits sequence	0111011010 0111011010 (W7)	0

From Table 12, it is clear that all the sequences strongly suggest the selection of a given outer encoded pattern.

If two decoded sequences show almost similar code distance and both suggest different legitimate paths/ trellis, then follow both the paths for a while and one of them will eventually end up making more error assumptions with respect to the ABS. We can even introduce a threshold based on the quantity of valid bits in the ABS, for some fix decoding window size, to make the outer decoding easier.

Sometimes an individual sequence may lead to an erroneous decoding path/trellis, and that is when the ABS becomes useful and serves as a retracting sequence. Likewise, the ABS alone may lead to an erroneous decoding path because of a lesser amount of valid bits within the selected decoding window. At this instance, one of the four decoded sequences may serve as a retracting sequence. This means the outer decoding algorithm has to be designed or modified in such a way that the ABS and other decoded sequences should retract each other during the decoding process. It is apparent from the previous discussion that decoding algorithms of selected conventional outer coding technique must be modified in order to rely more on the ABS in conjunction with any one decoded sequence. We can either use hard-decision decoding or soft-decision decoding for the outer code. To perform soft-decision decoding, we can assign higher weights for the valid accurate-bits and highly-accurate-bits positions, and consider the discarded-bits as it is or assign lesser weight.

5. CONCLUSION

The proposed coding technique provides a completely new and promising approach in the channel coding field. It is obvious from the above discussion that decoding complexity of the proposed coding technique is low as compared to currently used channel codes. The obtained results for the proposed codes as an inner code strongly indicate that proper teaming of outer and inner codes can yield a significant improvement in error correction performance with moderate decoding complexity. Any existing coding technique can be combined with the proposed codes after making some modifications to its decoding technique. These codes are of special interest in moderate to low signal-to-noise ratio applications.

REFERENCES

- [1] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423 and 623–656, 1948.
- [2] G. D. Forney, Jr., *Concatenated Codes*. Cambridge, MA: MIT Press, 1966.
- [3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," *Proc. 1993 Int. Conf. Commun. (Geneva)*, Switzerland, pp. 1064–1070, May 1993.
- [4] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Trans. On Commun.*, vol. 44, no. 10, pp. 1261-1271, October 1996.
- [5] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan. 1962.
- [6] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, "Practical loss-resilient codes," *Proceedings of the 29th annual ACM Symposium on Theory of Computing*, pp. 150-169, May 1997.
- [7] P. Oswald and A. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Trans. Inform. Theory*, vol. 48, pp. 3017-3028, Dec 2002.
- [8] S. Y. Chung, G. Forney, T. Richardson, and R. Urbanke, "On the Design of low-density parity check codes within 0.0045 db of the Shannon Limit," *IEEE Comm. Letters*, vol. 5, pp. 58-60, Feb 2001.
- [9] Y. Harada, Q. Wang, and V. Bhargava, "Performance of the Concatenated coding System with a Modified Erasure Declaration Viterbi Decoder," *IEEE Int. Conf. Commun., SUPERCOMM/ICC '94, Conference Record*, pp.1231-1234.
- [10] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *J. SIAM*, vol. 8, pp. 300–304, Jun. 1960.
- [11] J. Hagenauer and P. Hoher, "A Viterbi algorithm with soft-decision output and its applications," *IEEE GLOBECOM*, 1989, pp. 47.1.1-47.1.7.
- [12] T. Schaub and J. W. Modestino, "An erasure declaring Viterbi T. Schaub and J. W. Modestino, "An erasure declaring Viterbi decoder and its applications to concatenated coding systems," *IEEE Int. Conf. Commun., Toronto, Canada*, June 1986, pp. 1612-1616.
- [13] R. H. Deng and D. J. Costello, Jr., "High rate concatenated coding systems using bandwidth efficient trellis inner codes," *IEEE Trans. Commun.*, vol. COM-37, no.5, pp. 420-427, May 1989.
- [14] H. Yamamoto and K. Itoh, "Viterbi decoding algorithm for convolutional codes with repeat request," *IEEE Trans. Inform. Theory*, vol. IT-26., Sept. 1980, pp. 540-546.
- [15] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.
- [16] P. Elias, "Coding for noisy channels," *IRE Conv. Rec.*, pt. 4, pp. 37–46, Mar. 1955.
- [17] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, no. 4, pp. 260–269, Apr. 1967.

Author

Mahesh Patel received the B.S. degree in Electronics and Telecommunication Engineering from North Maharashtra University, India, and M.S. degree in Electrical Engineering from University of Texas at Arlington, United States of America in 2004 and 2007, respectively. He is currently working towards his PhD. degree in the Department of Electrical and Computer Engineering at the Prairie View A&M University, a member of Texas A&M University System. His current research interests include low rate channel codes, forward error correction codes and error estimation codes.



Dr. Annamalai is presently the Director of Center of Excellence for Communication Systems Technology Research, a Texas A&M Board of Regents approved University Research Center at the Prairie View A&M University, and a tenured professor in the Department of Electrical and Computer Engineering. He has over 16 years of research/teaching experience in wireless communications at Motorola, University of Victoria, Air Force Research Laboratory, Virginia Tech and PVAMU with over 200 peer-reviewed publications and 6 book chapters. Dr. Annamalai has been honored by his colleagues on numerous occasions for his excellence in research including winning the 2011 Roy G. Perry College of Engineering Outstanding Faculty (Research) Award, IEEE Leon Kirchmayer Prize Paper award, ASEE/AFOSR Summer Faculty Fellowships, NSERC Doctoral Prize, CAGS/UMI Distinguished Doctoral Dissertation Award, IEEE VTS/Motorola Daniel E. Noble Fellowship, among others. He had served on the Editorial Boards of four IEEE journals/transactions in the last 12 years, and has helped to organize a few major IEEE conferences on wireless communications including serving in the capacity of Technical Program Chair of the 2002 IEEE Vehicular Technology Conference in Vancouver, Canada. His current research interests include cooperative spectrum sensing, compressive sensing, cross-layer design for scalable multimedia transmission and cooperative wireless communications.

