Reduced Complexity Viterbi Decoding Based on the M-Algorithm and the Minimal Trellis

Daiana Nascimento Muniz
CPGEI
UTFPR
Curitiba-PR - Brazil
dai_muniz@yahoo.com.br

Franco A. dos S. Pezzini, Richard Demo Souza
DAELN - CPGEI
UTFPR
Curitiba-PR - Brazil
franco.augustus@gmail.com, richard@utfpr.edu.br

Cecilio Pimentel
CODEC - DES
UFPE
Recife-PE - Brazil
cecilio@ufpe.br

***Abstract*— In this paper we propose sub-optimum, reduced complexity decoding algorithms for convolutional codes. The algorithms are based on the minimal trellis representation for the convolutional code, and on the M algorithm. We analyze both the computational complexity, in terms of arithmetic operations, and the bit error rate performance of the proposed algorithms. Results demonstrate that large complexity reductions can be obtained while achieving a very good performance.**

I. Introduction

Digital communication systems make use of error correcting codes for increasing the realibility of the transmitted information [1]. Convolutional codes are among the most used ones, with extensive applications in modern wireless communication standards, such as WiMAX, EDGE, LTE. These codes have also been adopted in a series of applications that are energy constrained, as those involving biomedical implants and wireless sensor networks [2], [3].

The decoding of convolutional codes still demand a great amount of processing and energy consumption of a regular wireless digital receiver. For instance, in [4] it is shown that the decoding of a convolutional code, executed by the Viterbi algorithm (VA) [1], responds by 76% of the processing required by a HYPERLAN/2 receiver. In [5] the authors analyze different receiver implementations compatible to the IEEE 802.11 standard, showing that the VA contributes with 35% of the overall power consumption.

Thus, it is clear that complexity reductions in the decoding of convolutional codes would be of considerable use to a number of applications. A survey in the literature shows a series of related papers, among them we can cite [2], [6]–[15]. These works can be divided into three groups: i) hardware specific implementations [2]; ii) sub-optimum decoding methods [6]–[10]; iii) simpler trellis representations [11]–[15]. Supported by two new paradigms for the implementation of digital communications systems, the software defined radio model and the cognitive radio concept [16], we focus on software oriented implementations. Therefore, we consider only two of the above options: sub-optimum decoding methods and simpler trellis representations.

Most sub-optimum decoding algorithms were proposed based on the conventional trellis representation of a convolutional code and are similar to the VA. The sub-optimality comes from pruning some of the trellis edges, based on specific criteria. One of the most famous sub-otimum decoding algorithms is the M-algorithm (MA) [6], [7]. By its turn, the most known simpler trellis implementation approach is based on the work of McEliece and Lin [11], where the minimal trellis is defined. Such a trellis, even though of an irregular structure (the number os states and the number of edges emanating from each state is periodically time-varying) when compared to the conventional trellis, allows, in theory, for an optimal reduced complexity decoding.

In this paper we combine both approaches, sub-optimum decoding algorithms and simpler trellis representations. We design different decoding algorithms based on the MA operating over the minimal trellis. The basic idea is to define strategies to select the number of states with best metrics in each depth of the trellis. This number can be either fixed or variable when the MA operates over the minimal trellis. We analyze both the computational complexity, in terms of arithmetic operations, and the bit error rate (BER) performance for each proposed algorithm. Our results show that large reductions in complexity can be obtained while achieving a performance extremily close to that of the VA.

This paper is organized as follows. Some fundamental concepts are introduced in Section II. The new proposed algorithms are described in Section III. The BER performance of the new algorithms is numerically investigated in Section IV, while a complexity analysis is carried out in Section V. Finally, Section VI concludes the paper.

II. Fundamental Concepts

Consider a convolutional code $C(n,k,\nu)$, where ν, k and n are the overall constraint length, the number of binary inputs and binary outputs, respectively. The code rate is $R = k/n$.

Every convolutional code can be represented by a semi-infinite trellis which (apart from a short transient in its beginning) is periodic, the shortest period being called a *trellis module*. In general, a trellis module Φ for a convolutional code C consists of n' trellis sections, 2^{ν_t} states at depth t, 2^{b_t} edges emanating from each state at depth t, and l_t bits labeling each edge from depth t to depth $t+1$, for $0 \leq t \leq n'-1$.

In [11] a complexity measure, the trellis complexity, is proposed for comparing the computational effort per

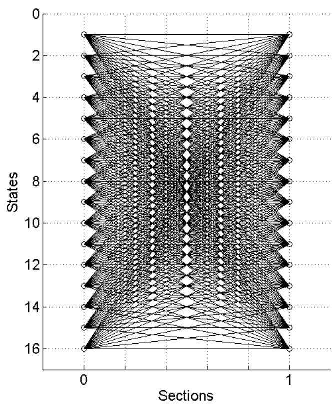

Fig. 1. Conventional trellis module for the $C(7,4,4)$ convolutional code with generator matrix in (2).

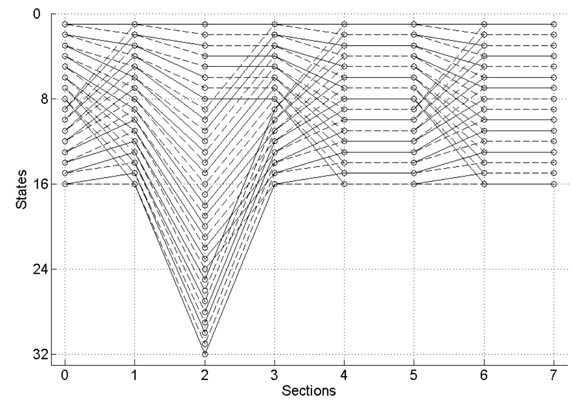

Fig. 2. Minimal trellis module for the $C(7,4,4)$ code in (2). Solid edges represent "0" codeword bits while dashed edges represent "1" codeword bits.

decoded bit of the VA operating over a given trellis module. The trellis complexity of the module Φ for the code C, denoted by $TC(\Phi)$, is [11]:

$$TC(\Phi) = \frac{1}{k}\sum_{t=0}^{n'-1} l_t\, 2^{\nu_t + b_t} \tag{1}$$

symbols per bit. In particular, the conventional trellis module Φ_{conv} for a rate $R = k/n$ convolutional code C consists of one trellis section with 2^ν initial states and 2^ν final states; each initial state is connected by 2^k directed edges to final states, and each edge is labeled with n bits. The trellis complexity of the conventional trellis is $TC(\Phi_{conv}) = (n/k)\,2^{\nu+k}$ symbols per bit. For instance, consider the $C(7,4,4)$ code with the following generator matrix:

$$G(D) = \begin{pmatrix} 1+D & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1+D & 1+D & D & 0 & 0 & 1 \\ D & D & D & 1 & 1 & 0 & 1 \\ 0 & D & 0 & D & D & 1+D & 1 \end{pmatrix} \tag{2}$$

whose conventional trellis module Φ_{conv} is shown in Fig. 1, with trellis complexity $TC(\Phi_{conv}) = 448$ symbols per bit. The limiting factor for using the VA is that its complexity grows large when the trellis module is dense, with many states and edges per states.

A. Minimal Trellis

The "minimal" trellis module, Φ_{min}, for convolutional codes was developed in [11]. This "minimal" structure has $n' = n$ sections and $l_t = 1$ bit per edge $\forall t$. The state complexity ν_t and the edge complexity b_t at depth t will be denoted by $\widetilde{\nu}_t$ and $\widetilde{b}_t$, respectively. The state and the edge complexity profiles of the "minimal" trellis module are denoted by $\widetilde{\nu} = (\widetilde{\nu}_0, \ldots, \widetilde{\nu}_{n-1})$ and $\widetilde{b} = (\widetilde{b}_0, \ldots, \widetilde{b}_{n-1})$, respectively. It has been shown in [11] that for many convolutional codes the trellis complexity $TC(\Phi_{min})$ of the "minimal" trellis module is considerably smaller than the trellis complexity $TC(\Phi_{conv})$ of the conventional trellis module.

The minimal trellis presents an irregular pattern in each section. For instance, Fig. 2 shows the minimal trellis module Φ_{min} for the $C(7,4,4)$ code with generator matrix in (2). While the single-section conventional trellis module Φ_{conv} in Fig. 1 has a very regular structure, 16 states with 16 edges leaving each state, each edge labeled by 7 bits, the minimal trellis module Φ_{min} in Fig. 2 has $n = 7$ sections, with 16 or 32 states each. Note that only the first, second, fourth and sixth sections have information bits, i.e., two edges leave each state ($\tilde{b}_t = 1$).

The complexity of the minimal trellis module Φ_{min} is:

$$TC(\Phi_{min}) = \frac{1}{k}\sum_{t=0}^{n-1} 2^{\widetilde{\nu}_t + \widetilde{b}_t} \tag{3}$$

symbols per bit. For the $C(7,4,4)$ code, $\widetilde{\nu} = (4,4,5,4,4,4,4)$ and $\widetilde{b} = (1,1,0,1,0,1,0)$, thus $TC(\Phi_{min}) = 48$ symbols per bit, while $TC(\Phi_{conv}) = 448$ symbols per bit. Therefore, the complexity of the minimal trellis for this code is only 10.7% of the complexity of the conventional one.

B. M-Algorithm

Another approach for reducing the decoding complexity is the utilization of a sub-optimum algorithm such as the MA [6]. The MA is very similar to the VA, the difference being in the fact that the MA calculates the accumulative metric of only $M \leq 2^\nu$ paths along the conventional trellis module. The MA can be described by the following steps:

1) *Start from the leftmost module of the trellis.*
2) *Expand all the states metrics stored at depth $t-1$ to depth t. Select the surviving edges for each state reached at time t.*
3) *Select the M states at time t with the best metrics, discard the others.*
4) *Store the M selected states, their metrics and surviving edges.*

5) *Repeat steps 2-4 for all trellis modules.*
6) *Estimate the transmitted sequence by tracebacking from the state with the best final metric.*

Note, from the above description, that the MA selects the M best states before expanding the metrics of these states. Besides that, there is no difference between the MA and the VA. The MA can achieve bit error rates close to that of VA with reduced complexity [10]. It is also used when the convolutional code has a large overall constraint length [10].

III. MA over the Minimal Trellis

When the MA operates over the conventional trellis, the parameter M is fixed, however, since the minimal trellis has a time-varying state profile, the number of states selected in each section of this trellis can be either fixed or variable. We propose three different variants of the MA to operate over the minimal trellis, namely, Modular M-Algorithm (MMA); Proportional M-Algorithm (PMA); Fixed M-Algorithm (FMA). These algorithms differ only in step 3), that is, in the way they select the best states.

The MMA selects the M states only in the end of each minimal trellis module. Thus, in the other sections there is no selectio of the best states. The metrics are expanded up to the end of the module, storing the metrics of all states reached by any surviving edge. Then, step 3) becomes:

3) *If this is the last section of the module, store the M best states and discard the others; Otherwise store the metrics of all states reached by a surviving edge.*

In the PMA the number of stored states varies according to the number of states in that section. The number of states stored per section is $2^{\nu_t} \times \frac{M}{2^{\nu}}$. Recall that 2^{ν_t} is the number of states at that section of the minimal trellis, while 2^{ν} is the number of states in the conventional trellis. For instance, if in the MA operating over the conventional trellis the parameter M means a reduction of 50% in the number of stored states, then in the PMA this value of M means a reduction of 50% in the number of states stored in each section of the minimal trellis. Thus, the number of stored states is proportional to the number of states in that section of the minimal trellis. Step 3) of the PMA can be written as:

3) *Store the $2^{\nu_t} \times \frac{M}{2^{\nu}}$ states with the best metric for each section. Discard the others.*

In the FMA only M states are stored at each section of the minimal trellis, independent of the trellis pattern or section. Step 3) of the FMA can be described as:

3) *Store the M best states at each section. Discard the others.*

IV. BER Performance

In this section we numerically investigate the BER performance of the proposed algorithms. In the simulations we generated 30000 blocks of 300 information bits. The coded blocks were then BPSK modulated and sent over the AWGN channel. We considered the $C(7,4,4)$ code with generator matrix in (2) and the $C(3,2,4)$ code with generator matrix:

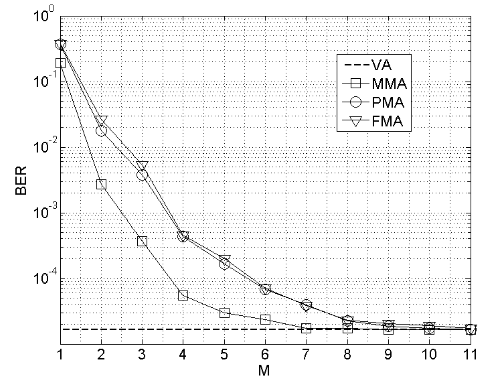

Fig. 3. BER versus M for the proposed algorithms, considering the $C(7,4,4)$ code, soft decision decoding, and $E_b/N_0 = 6.0$ dB.

$$G(D) = \begin{pmatrix} 1+D^2 & 1 & D \\ D & 1+D+D^2 & 1+D \end{pmatrix}. \quad (4)$$

In Figure 3, which considers the $C(7,4,4)$ code and $E_b/N_0 = 6.0$ dB, we can see that the MMA requires $M = 7$ to achieve the same BER performance of the VA, which is around 10^{-5}. PMA and FMA require larger values, $M = 10$ and $M = 11$, respectively. Thus, MMA needs to consider a smaller number of surviving sates than the PMA and the FMA. But recall that MMA only selects the surviving states at the end of the module, while the PMA and the FMA select them at each section. A different result is obtained for the $C(3,2,4)$ code at an $E_b/N_0 = 6.5$ dB (results not shown due to the lack of space). In this case the MMA needs $M = 6$ while the PMA and FMA required $M = 8$ to approach the same BER close to 10^{-5} as the VA. The PMA and the FMA have exactly the same performance because the minimum trellis module for this code has a constant number of states per section, namely 16 states per section.

V. Complexity Analysis

The complexity of a decoding algorithm can be determined as a function of the number of arithmetic operations required by the algorithm. In this paper we consider only summations (S), multiplications (X), and comparisons (C). First we consider only the operations required to calculate the accumulated state metrics. Later we take into account the effort required to select the best M states, when needed by the MA and its variants. Memory reads and writes are not taken into account.

Either in the VA, in the MA, or in the proposed algorithms, the first step in decoding is to calculate the edge metrics. These metrics may be calculated by means of the Hamming distance (hard decision) or the Euclidian distance (soft decision). Supposing the use of the conventional trellis, for each group of n received bits the edge metrics are calculated with respect to the labels of the edges connecting states in the module. Assumming the use of a constant modulus modulation such as BPSK and soft decision decoding, the minimum number of operations required for calculating each edge metric is $nX+(n-1)S$. Then, the state metrics are expanded, by summing the previous state metrics at the left of the module, with the calculated edge metrics. The new state metrics are determined by comparing the value of the expanded metrics that reach a given state, and storing only the best ones. If N_e^i edges reach state i, then the update of this state metric requires $N_e^i S+(N_e^i-1)C$ operations. If the number of states reached by at least one edge in the end of the module is N_s, then the total number of operations Λ_{conv} in the conventional trellis module Φ_{conv}, required to calculate the accumulated state metrics, is:

$$\Lambda_{conv}=\sum_{i=1}^{N_s} n\ N_e^i(X+S)+(N_e^i-1)C. \qquad (5)$$

In the case of the VA, $N_e^i=2^k\ \forall i$ and $N_s=2^\nu$. For the MA, N_e^i and N_s are random variables.

The complexity of the decoding algorithms over the minimal trellis can be calculated in an analogous way, with few differences. First, every edge is labeled by only one bit. Second, the analysis is carried out over the n sections. Thus, the number of operations Λ_{min} considering the minimal trellis is:

$$\Lambda_{min}=\sum_{t=0}^{n-1}\sum_{i=1}^{N_s^{t+1}} N_e^{it}\,(X+S)+(N_e^{it}-1)\,C, \qquad (6)$$

where N_e^{it} is the number of edges starting in section t and reaching state i in section $t+1$, and N_s^{t+1} is the number of states in section $t+1$ that are reached by edges coming from section t. Note that these parameters are a function of the state and edge complexity profiles of the minimal trellis module, and of the type of algorithm in use. For the VA over the minimal trellis $N_e^{it}=\frac{2^{(b_t+\nu_t)}}{2^{\nu_{t+1}}}$ and $N_s^{t+1}=2^{\nu_{t+1}}$.

In the VA the number of operations per trellis module is constant. However, in the sub-optimum algorithms the number of comparisons varies, since the number of edges reaching a given state is a random variable, as well as the number of states reached at each trellis section. These random variables are tracked during the computer simulations, so that the actual average values can be used for calculating the computational complexities. For instance, Table I lists $\bar{N}_s^{t+1}$, the average number of states in section $t+1$ that are reached by edges coming from section t, for the three proposed algorithms, the $C(7,4,4)$ code, $E_b/N_0=6.0$ dB, and using the minimum M so that the BER performance approaches that of the VA ($M=7$ for the MMA, $M=10$ for the PMA and $M=11$ for the PMA). Table II shows similar data but for the $C(3,2,4)$ code.

The average number of edges starting in section t and reaching state i in section $t+1$, $\bar{N}_e^{it}$, can be written as a function of $\bar{N}_s^t$, such that for the MMA we have $\bar{N}_e^{it}=\frac{2^{b_t}\bar{N}_s^t}{\bar{N}_s^{t+1}}$, for the PMA

$$\bar{N}_e^{it}=\frac{2^{b_t+\nu_t-\nu}\min\{M,\bar{N}_s^t\}}{\bar{N}_s^{t+1}},$$

TABLE I
$\bar{N}_s^{t+1}$ FOR $C(7,4,4)$ AND $E_b/N_0=6.0$ DB.

Algorithm	**Section** t						
	0	1	2	3	4	5	6
MMA ($M=7$)	12.1	24.1	14.9	15.9	15.9	16.0	16.0
PMA ($M=10$)	14.6	19.8	13.6	15.8	10.0	15.7	10.0
FMA ($M=11$)	14.8	21.8	9.8	15.5	11.0	15.9	11.0

TABLE II
$\bar{N}_s^{t+1}$ FOR $C(3,2,4)$ AND $E_b/N_0=6.5$ DB.

Algorithm	**Section** t		
	0	1	2
MMA ($M=6$)	10.7	15.8	15.8
PMA ($M=8$)	11.0	13.8	8.0
FMA ($M=8$)	11.0	13.8	8.0

while for the FMA

$$\bar{N}_e^{it}=\frac{2^{b_t}\min\{M,\bar{N}_s^t\}}{\bar{N}_s^{t+1}}.$$

The minimum between M and $\bar{N}_s^t$ appears because, during the operation of the FMA or the PMA, $\bar{N}_s^t$ can be smaller than M (this is illustrated in Table I for the FMA). Then, defining $M_t^\star=\min\{M,\bar{N}_s^t\}$, we can equate the complexity of each individual algorithm as:

$$\Lambda_{min}^{FMA}=\sum_{t=0}^{n-1}2^{b_t}M_t^\star\,(X+S)+(2^{b_t}M_t^\star-\bar{N}_s^{t+1})\,C, \qquad (7)$$

$$\Lambda_{min}^{PMA}=\sum_{t=0}^{n-1}2^{b_t+\nu_t-\nu}M_t^\star\,(X+S)+(2^{b_t+\nu_t-\nu}M_t^\star-\bar{N}_s^{t+1})\,C, \qquad (8)$$

$$\Lambda_{min}^{MMA}=\sum_{t=0}^{n-1}2^{b_t}\bar{N}_s^t\,(X+S)+(2^{b_t}\bar{N}_s^t-\bar{N}_s^{t+1})\,C. \qquad (9)$$

Moreover, since the number of states reached at any trellis section varies, the effort to select the best states (when required) also varies. From Table I we can see that when running the FMA, for instance, in the first section the best 11 states have to be selected out of around 15 (N_s^1 = 14.8 in Table I), in the second section 11 states have to be selected out of about 22 (N_s^2 = 21.8), in the third section no selection is needed since less than $M=11$ states are reached in average (N_s^3 = 9.8), and so on. Such an effort, in terms of comparisons, of selecting the y largest or smallest elements within a vector of z elements can be approximated by [17]:

$$\Pi(y,z)=z-y+\sum_{i=z+1-y}^{z}\log_2 i. \qquad (10)$$

Therefore, in order to fairly compare the proposed algorithms, we have to take into account the effort required to select the best states, an action carried out at each section

TABLE III
ARITHMETIC OPERATIONS REQUIRED FOR DECODING $C(7,4,4)$.

Algorithm	Operations	Overall
VA_c	$1792X + 1792S + 240C$	3824
VA_m	$192X + 192S + 64C$	448
MMA ($M = 7$)	$155.63X + 155.63S + 72.27C$	383.53
PMA ($M = 10$)	$119.94X + 119.94S + 131.77C$	371.65
FMA ($M = 11$)	$118.44X + 118.44S + 152.06C$	388.94

TABLE IV
ARITHMETIC OPERATIONS REQUIRED FOR DECODING $C(3,2,4)$.

Algorithm	Operations	Overall
VA_c	$192X + 192S + 48C$	432
VA_m	$80X + 80S + 32C$	192
MMA ($M = 6$)	$49.2X + 49.2S + 35.9C$	134.3
PMA ($M = 8$)	$40.0X + 40.0S + 51.1C$	131.1
FMA ($M = 8$)	$40.0X + 40.0S + 51.1C$	131.1

in the PMA and FMA and only at the end of the module in the MMA. Then, the complexity of the proposed algorithms over the minimum trellis could be written as:

$$\Lambda_{min}^{total} = \Lambda_{min} + \sum_{t=0}^{n-1} \Pi_t \tag{11}$$

where Λ_{min} is calculated according to (7-9), and Π_t is the average number of comparisons required at section t to select the best states, according to (10), and as function of M, $\bar{N}_s^{t+1}$, and to the particular operation of the algorithm.

Based on (11) we can present the average number of arithmetic operations required by each of the proposed algorithms, considering the minimum value of M that allows the same performance as that of the VA. The case of the $C(7,4,4)$ code is shown in Table III, while Table IV deals with the case of the $C(3,2,4)$ code. Besides listing the number of additions, multiplications and comparisons, the table also shows the overall number of operations supposing that all three have the same cost. That is reasonable to assume specially in the case of BPSK modulation, where the multiplications are only times $+1$ or -1. The number of operations required by the VA, over the conventional and minimum trellises, are also shown.

From the tables, since all algorithms provide the same BER performance at those E_b/N_0 values, we can conclude that PMA would be the best choice for the case of the $C(7,4,4)$ code, with MMA performing second. For the $C(3,2,4)$ code any of the three algorithms would be the a good option, even though FMA and PMA slightly outperform MMA. This shows that the best algorithm depends on the code in use. Moreover, the relations might change if the cost of the individual operations are not equal. From the tables we can also see that considerable additional savings can be obtained from the use of the proposed algorithms, when compared to the savings already resultant from the use of the VA over the minimal trellis. Such additional savings go from around 17%, in the case of the $C(7,4,4)$ code, to more than 30% in the case of the $C(3,2,4)$ code.

VI. FINAL COMMENTS

In this paper we proposed sub-optimum decoding algorithms operating over the minimal trellis. The algorithms are based on the MA, and their operation is matched to characteristics inherent to the minimal trellis. Three algorithms were proposed. Numerical results showed that the same BER performance obtained by the optimum decoders can be achieved by the sub-optimum proposed algorithms with much reduced computational complexity.

VII. ACKNOWLEDGMENTS

This work was partially supported by CNPq under grants 561443/2008-4 and 132078/2009-0.

REFERENCES

[1] S. Lin and D. J. Costello, *Error Control Coding*, 2nd edition, Prentice-Hall, 2004.

[2] B. Tomatsopoulos and A. Demosthenous, "A CMOS hard-decision analog convolutional decoder employing the MFDA for low-power applications," *IEEE Trans. Circuits Syst. I: Regular Papers*, vol. 55, no. 9, pp. 2912-2923, Oct. 2008

[3] E. Shih, S. Cho, F. S. Lee, B. H. Calhoun and A. Chandrakasan, "Design considerations for energy-efficient radios in wireless microsensor networks," *Journal of VLSI Signal Processing*, vol. 37, pp. 77- 94, 2004.

[4] K. Masselos, S. Blionas, and T. Rautio, "Reconfigurability requirements of wireless communication systems," *Proc. IEEE Workshop on Heterogeneous Reconfigurable Systems on Chip*, April 2002.

[5] B. Bougard et al, "Energy-scalability enhancement of wireless local area network transceivers," *Proc. IEEE Workshop on Signal Processing Advances in Wireless Communications*, July 2004.

[6] K. S. Zigangirov and V. D. Kolesnik, "List decoding of trellis codes," *Probl. Contr. Inform. Theory*, vol. 6, pp. 347–364, 1980.

[7] J. B. Anderson, J.B. "Limited search trellis decoding of convolutional codes," *IEEE Trans. Inform. Theory*, vol. 35, no. 5, pp. 944-955, Sept. 1989.

[8] F. Chan and D. Haccoun, "Adaptive viterbi decoding of convolutional codes over memoryless channels," *IEEE Trans. Commun.*, vol. 45, no. 11, pp. 1389–1400, November 1997.

[9] R. Henning and C. Chakrabarti, "An approach for adaptively approximating the Viterbi algorithm to reduce power consumption while decoding convolutional codes," *IEEE Trans. Signal Proc.*, vol. 52, no. 5, pp. 1443-1451, May 2004.

[10] G. J. Pottie and D. P. Taylor, "A comparison of reduced complexity decoding algorithms for trellis codes," *IEEE J. Selected Areas in Commun.*, vol. 7, no. 9, pp.1369-1380, Dec. 1989.

[11] R. J. McEliece and W. Lin, "The trellis complexity of convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, n. 6, pp. 1855-1864, Nov. 1996.

[12] B. Uchôa-Filho, R. D. Souza, C. Pimentel, and M. Jar, "Convolutional codes under minimal trellis complexity measure," *IEEE Transactions on Communications*, vol. 57, pp. 1-5, Jan. 2009.

[13] C. Pimentel, R. D. Souza, B. F. Uchôa-Filho, M. E. Pellenz, "Generalized punctured convolutional codes with unequal error protection," *EURASIP Journal on Advances in Signal Processing*, 2008.

[14] A. Katsiotis, P. Rizomiliotis, and N. Kalouptsidis, "New constructions of low complexity convolutional codes," *in Proc. IEEE Int. Conf. on Commu. (ICC 2009)*, Dresden, Germany, 2009.

[15] F. Hug, I. Bocharova, R. Johannesson, and B. D. Kudryashov, "Searching for high-rate convolutional codes via binary syndrome trellises, *in Proc ISIT 2009*, Seoul, Korea, 2009, pp. 1358-1362.

[16] J. Mitola, *Cognitive Radio: An integrated Architecture for Software Defined Radio*, Ph.D. Dissertation, KTH, Stockholm, Sweden, Dec. 2000.

[17] D. Knuth, *The Art of Computer Programming, Volume 3: Sorting and Searching*, Third Edition. Addison-Wesley, 1997.