On BCJR State Metric Quantization for Turbo Equalization

Brian M. Kurkoski, Kazuhiko Yamaguchi and Kingo Kobayashi
Dept. of Information and Computer Engineering
University of Electro-Communications
Chofu, Tokyo, Japan
Email: {kurkoski,yama,kingo}@ice.uec.ac.jp

***Abstract*— Vector quantization of the BCJR and Viterbi algorithms' state metrics for detection of finite-state channels is considered. An estimate is given for the gain associated with vector quantization, over conventional implementations. This is expressed using the volume of the recurrent region, and the maximum state metric difference, which are both intrinsic properties of the channel detector. One application of this gain is the complexity evaluation of a previously proposed lookup-table BCJR implementation. The BCJR algorithm is of interest in turbo equalization used for communication over intersymbol-interference and finite-state Markov channels.**

I. Introduction

The Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm is a key component in turbo equalization, a technique which can be used to approach the capacity of channels with memory. Intersymbol interference (ISI) channels and finite-state Markov (FSM) channels, such as the Gilbert-Elliot channel, are used as models in applications such as wireless and wireline communications, magnetic and optical recording. Turbo equalization has been well-studied for ISI channels [1] [2]. Using turbo codes and low-density parity check (LDPC) codes to communicate over FSM channels, such as the Gilbert-Elliot channel, has also attracted research interest [3] [4] [5] [6]; proposals often use the BCJR algorithm to detect the channel state, and can be viewed as an instance of turbo equalization. Analysis and performance of turbo equalization often relies on the BCJR algorithm, and it is desirable to have an understanding of the algorithm's state metrics.

The BCJR algorithm operates on an M-state trellis and it computes forward and backward state metrics which nominally are scalars. In this paper, we consider the state metrics as an M (or $M-1$) dimensional vector. As the state metric recursion continues forward, the state metric vector has a particular property: there is a distinct region of the vector space which supports the state metrics. This region of reachable space is called the recurrent region, and has been studied for certain ISI channels [7].

We propose a search technique which finds the rectangular lattice points inside the recurrent region. This method can be applied in a straightforward manner to arbitrary trellises, and uses quantization of the transition metrics. The technique of [7], although applicable for continuous transition metrics, requires extensive calculations.

The characterization of the recurrent region is of interest for analyzing a lookup-table based implementation of the BCJR algorithm [8]. In this implementation, only those state metrics inside the recurrent region are used in the lookup table. We will use this characterization to estimate the lookup table size. Vector quantization is also useful for analytical techniques which approximate the true BCJR algorithm by using quantized state metrics.

Section II gives an overview of the system under consideration, and describes the BCJR algorithm as it is used to detect ISI and FSM channels. Section III defines the recurrent region, and gives the proposed technique by which the set of rectangular lattice points inside the recurrent region can be found. Also given is a bound on the gain associated with vector quantization. Section IV considers numerical results for the proposed technique, and an application with the implementation of the lookup table algorithm. Finally, Section V gives the conclusions.

II. System Model and BCJR algorithm

A. System Model

We assume that an error-correcting code with a suitable soft-input, soft-output decoder, such as a LDPC or turbo code, is used to communicate over an ISI or FSM channel. At the receiver, turbo equalization is performed by the BCJR detector and an appropriate decoder iteratively sharing soft information. The transmitted codeword sequence is x_k, $k = 1, \cdots, N$, for a code of block length N. The received symbol sequence is y_k. The detector-to-decoder message is U_k, a log-probability value. The decoder-to-detector message is $G_k = \log P(x_k = 1)/P(x_k = 0)$.

The ISI channel model has binary input and real output, and a causal impulse response $h(D) = \sum_{i=0}^{\nu} h_i D^i$, where D is the delay operator, and ν is the length of the channel interference, so that the number of states is $M = 2^\nu$. The noise-free channel output is $c(D) = h(D)x(D)$. The receiver observes $y_k = c_k + n_k$, where n_k is memoryless noise with known distribution.

For the FSM channel, data is transmitted over one of M binary symmetric channels; the channel is chosen according

This research was partially supported by the Ministry of Education, Science, Sports and Culture; Grant-in-Aid for JSPS Fellows, number 00004908, 2004; and Grant-in-Aid for Scientific Research (C) number 16560324, 2004.

to a hidden Markov model with M states, with transition probability from state i at time k, to state j at time $k+1$ is given by the matrix $P = \{p_{ij}\}$. The crossover error probability for binary symmetric channel m, $m = 0, \cdots, M-1$ is given by $Q = q_m$. The error pattern is z_k, so that the received sequence is $y_k = x_k + z_k$, modulo 2. It is assumed that the Markov model is stationary and irreducible. If $M = 2$, the FSM is the Gilbert-Elliot channel model.

Both channels can be described by a trellis section, with state $s_k, s_{k+1} \in \{0, \cdots, M-1\}$ at time $k, k+1$. For each trellis edge e, let $s^S(e)$, $s^E(e)$, represent the starting and ending state, respectively; for the ISI model, $x(e)$ and $y(e)$ represent the input and output edge labels. For the FSM model, there are parallel transitions between states, corresponding to the possible error patterns $z_k = \{0, 1\}$.

B. BCJR Algorithm

Both the ISI channel and the FSM channel can be detected using the BCJR algorithm. For the general model, the transition metrics using edge notation are:

$$H_k(e) \quad = \quad -G_k(x(e)) - \log P(y_k|e) - \log p_{s^S(e),s^S(e)}.$$

For the ISI model, the term $P(y_k|e)$ is interpreted to mean $P(y_k|y(e))$, and this can be computed from the known noise distribution of the channel. Further, $p_{s^S(e),s^S(e)}$ is a constant which does not affect the output of the algorithm, and so can be dropped. For the FSM channel, the term $P(y_k|e)$ is interpreted to mean $P(y_k|x(e), s^S(e), s^E(e))$, and the channel error probability Q is sufficient to calculate this term.

Define the $\min^*()$ "min-star" operation as:

$$\min^*(A, B) \quad = \quad \min(A, B) - \log(1 + e^{-|A-B|}).$$

The $\min^*()$ operation is associative, so operations on more than two arguments are well defined.

The forward BCJR state metrics at time k for state m, $A_k(m)$, are recursively computed as:

$$A_{k+1}(m) \quad = \quad \min^*_{e:s^E(e)=m}(H_k(e) + A_k(s^S(e))). \quad (1)$$

If the min operation replaces the $\min^*$ operation, then (1) becomes the Viterbi state metric recursion [9]. This representation, rather than the usual max*-log-MAP [10], permits ready comparison with results for Viterbi detection of ISI channels, for which there is a wider variety of known results. For low channel noise, the $\min^* \approx \min$ approximation becomes good.

Consider the state metrics at time k as a vector:

$$\mathbf{A}_k \quad = \quad (A_k(0), \cdots, A_k(M-1)).$$

The state transitions H_k depend upon the algorithm inputs G_k and y_k. Then, the forward state metric recursion (1) can be represented as a function f_A:

$$\mathbf{A}_{k+1}(m) \quad = \quad f_A(\mathbf{A}_k, y_k, G_k).$$

Let the similar Viterbi state metric recursion function be $f^{\mathrm{v}}_A(\mathbf{A}_k, y_k, G_k)$.

The backward state metrics $B_k(m)$ are computed in a similar fashion. The *a posteriori* output of the BCJR algorithm is U_k.

C. Quantization of State Metrics

In a circuit implementation of the BCJR or Viterbi algorithm, the state metrics $A_k(m)$ are scalar quantized, but often quantizers are determined ad hoc. An analytical approach to the problem is to find the maximum difference between any two state metrics; this bound can be used to determine the appropriate dynamic range of the state metric quantizer. This has been studied for certain cases such as Viterbi detection of ISI channels [11], and BCJR [12] and Viterbi [13] decoding of convolutional codes.

Let $\Delta_{i,j}$ and $\Delta^{\mathrm{v}}_{i,j}$ be the maximum state metric difference between states i and j at any time k, for the BCJR and Viterbi algorithm state metrics, respectively. Let Δ^* be the maximum state metric difference for all BCJR state metric differences, $\Delta^* = \max_{i,j} \Delta_{i,j}$. Similarly, $\Delta^{*,\mathrm{v}}$ is the maximum difference for all of the Viterbi state metric differences. It is sufficient for a BCJR implementation to quantize the state metrics in the range $(0, \Delta^*)$. However it is common in practice to quantize the state metrics in the range $(0, 2\Delta^*)$. The state metrics are permitted to overflow and a normalization step is avoided, but an additional bit of storage per state metric is required.

Thus a *conventional implementation* refers to the BCJR or Viterbi algorithm quantizing the M state metrics $A_k(m), m = 0, \cdots, M-1$ individually, using scalar quantization. Note that the quantity $\mathbf{A}_k$ is a vector, and so the conventional quantization scheme is rectangular quantization of the vector quantity $\mathbf{A}_k$.

D. Table Lookup Implementation

Previously, we proposed a lookup-table implementation of the BCJR algorithm [8]. The nominal complexity is three table lookup operations per bit, one lookup each for the forward recursion, backward recursion and the computation of the *a posteriori* output.

The inputs y_k and G_k are quantized to $\hat{y}_k$ and $\hat{G}_k$, respectively, and the quantized state metrics are $\widehat{\mathbf{A}}_k$. The forward recursion is implemented by a single lookup table $f_{A,Q}$:

$$\widehat{\mathbf{A}}_{k+1} \quad = \quad f_{A,Q}(\widehat{\mathbf{A}}_k, \hat{y}_k, \hat{G}_k).$$

This implementation has lower complexity and requires less state metric storage than the conventional approach. However, the tradeoff is the size of the lookup table required. An application of the results in this paper is an analytic determination of the lookup table size.

III. Recurrent Regions

A. Definition and Search Technique

The region of space which can be occupied by the state metric vector is called the recurrent region. Much as convolutional codes and ISI channels have an distance spectrum which is an intrinsic property of the trellis, we also regard the recurrent region as an intrinsic trellis property.

Definition If at some fixed time the state metrics are initialized to $\mathbf{A} = (0, 0, \ldots, 0)$, then the *forward BCJR recurrent region* is the set of state metrics which can be reached by

an arbitrary number of forward recursion steps of the BCJR algorithm. Similarly, the *forward Viterbi recurrent region* is the set of state metrics which can be reached by an arbitrary number of forward recursion steps of the Viterbi algorithm.

Backward recurrent regions also exist, and are distinct from the forward recurrent regions. They are similarly defined to the forward regions, and time is allowed to be negative.

The inputs y_k and G_k are bounded between a specified maximum and minimum value, $y_{\min}, y_{\max}, G_{\min}, G_{\max}$. If these inputs are bounded, then the recurrent region is finite. The recurrent region does not depend upon the distribution of y_k and G_k, but it does depend upon the non-zero range.

We propose a technique which searches for the set of rectangular lattice points with spacing δ which are inside the Viterbi recurrent region. Let this set of lattice points be $\mathcal{A}$.

Recurrent Region Search Technique

1) Choose a starting step size δ_0 and initialize a search set vector $\mathcal{R}_y = \{y_{min}, y_{min} + \delta_0, y_{min} + 2\delta_0, \cdots, y_{max}\}$ and $\mathcal{R}_G = \{G_{min}, G_{min} + \delta_0, G_{min} + 2\delta_0, \cdots, G_{max}\}$.
2) Let $\mathcal{A}_i$ be the candidate set of lattice points on iteration i of the algorithm. Initialize $\mathcal{A}_0 = \{(0, 0, \ldots, 0)\}$.
3) Find the set $\mathcal{A}'_i = \{f^{\mathrm{v}}_A(\mathbf{A}, y, G)\}$ for all $\mathbf{A} \in \mathcal{A}_{i-1}, y \in \mathcal{R}_y, G \in \mathcal{R}_G$.
4) Update $\mathcal{A}_i = \mathcal{A}'_i \bigcup \mathcal{A}_{i-1}$.
5) If $\mathcal{A}_i = \mathcal{A}_{i-1}$, then stop, $\mathcal{A} = \mathcal{A}_i$ is the set of lattice points inside the recurrent region. Otherwise, go to step 3.

If the arguments of the $\min()$ function, including $\mathcal{R}_y$ and $\mathcal{R}_G$, are rational with denominator d, then the state metrics are also rational numbers with denominator d. In this way, the Viterbi state metric vectors will be points on a rectangular lattice with spacing δ which is an integral multiple $1/d$.

This algorithm is effective for finding the Viterbi algorithm's recurrent region for a specified trellis. The $\min()$ function approximates the $\min^*()$ function, and empirical evidence suggests that the Viterbi recurrent region is a good approximation of the BCJR recurrent region at signal-to-noise ratios of interest.

To illustrate the recurrent region search technique, consider the combination of the even-mark modulation constraint, which permits ones to occur only in pairs, and the partial-response class one channel, $h(D) = 1 + D$. The combined trellis for this EMM-PR1 channel has three states. Fig. 1 shows the lattice points with spacing $\delta = 0.5$ within the EMM-PR1 recurrent region. In the legend, $\mathcal{A}_i \setminus \mathcal{A}_{i-1}$ denotes the points added on iteration i. The search terminated after iteration number 7. Also shown is the recurrent region for continuous transition metrics [7], labelled "bound."

A constant can be added to the state metrics without affecting the output of the BCJR or Viterbi algorithms. The state metric $A_k(M-1)$ is normalized to zero; thus $\mathcal{A}$ can be considered to be a set of points in an $(M-1)$-dimensional space.

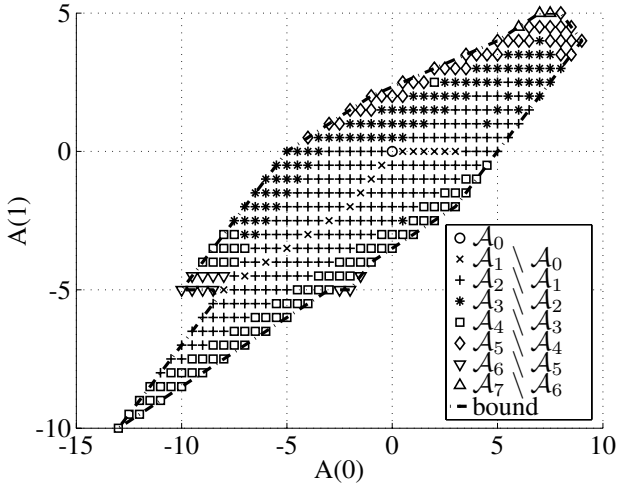

Fig. 1. Recurrent region search technique.

B. Volume of the Recurrent Region

The recurrent region has a volume $\mathcal{V}$, which is independent of the selected quantizer. We can bound this volume $\mathcal{V}$ from above by observing that each point in $\mathcal{A}$ is an element of a rectangular lattice with spacing δ, and is the center of a $(M-1)$-dimensional cube with side length δ. This cube has volume δ^{M-1}. The recurrent region is a subset of the union of these cubes, since the cubes on the boundary of the recurrent region are partially outside of the recurrent region. The total volume of these cubes forms an upper bound on $\mathcal{V}$:

$$\mathcal{V} \quad \leq \quad |\mathcal{A}|\, \delta^{M-1}. \tag{2}$$

C. Gains by Recurrent Region Quantization

By quantizing only the recurrent region, it is possible to reduce the amount of state metric quantization required relative to conventional implementations. Here, we estimate this gain, measured in the number of quantization bits per state metric.

For the conventional implementation, assume that M metrics are scalar quantized between $(0, 2\Delta^*)$, using uniform spacing between codepoints of δ. Thus, $\log_2(2\Delta^*/\delta)$ bits of quantization are required, per state.

On the other hand, $\mathcal{A}$ represents the intersection of the recurrent region and the $(M-1)$-dimensional rectangular lattice with spacing δ. A vector quantizer using $\mathcal{A}$ has the same distortion as the conventional quantizer, and uses $(1/M)\log_2|\mathcal{A}|$ bits per state metric.

Let n_b^{gain} represent the number of bits per state metric gained by quantizing only the recurrent region. Then,

$$\begin{aligned} n_b^{gain} &= \log_2(2\Delta^*/\delta) - (1/M)\log_2|\mathcal{A}|, \\ &= \log_2\left(\frac{2\Delta^*}{\delta\ \sqrt[M]{|\mathcal{A}|}}\right). \end{aligned}$$

By applying the volume bound (2):

$$n_b^{gain} \quad \leq \quad \log_2 \frac{2\Delta^*}{\sqrt[M]{\mathcal{V}}} - \log_2 \sqrt[M]{\delta}.$$

TABLE I

QUANTIZATION GAINS ASSOCIATED WITH VARIOUS FINITE-STATE CHANNELS, WHEN VECTOR QUANTIZATION IS USED.

Channel	M	Recurrent Region Volume Bound $\mathcal{V}$	Max State Metric Difference $\Delta^{*,\mathrm{v}}$	Gain n_b^{gain}	approximate Forward Recursion Table Size
EMM-PR1	3	91	13	2.5 bits/state	$2^{15.5} \times 11$ bits (62 kbytes)
PR2	4	210	21	3.4 bits/state	$2^{15.4} \times 11$ bits (58 kbytes)
EPR4	8	40107	24	3.7 bits/state	$2^{23.4} \times 19$ bits (25 Mbytes)
SI	3	30	19	3.6 bits/state	$2^{13.2} \times 8$ bits (9 kbytes)

Thus, the gain is separated into two portions, one which depends upon intrinsic properties of the trellis, and the other which depends upon the quantization scheme selected. In the limit $\delta \to 0$, the volume approximation becomes exact and:

$$n_b^{gain} \approx \log_2 \frac{2\Delta^*}{\sqrt[M]{\mathcal{V}}}. \quad (3)$$

In this case, the gain associated with quantizing only the recurrent region depends only on the intrinsic trellis properties.

IV. NUMERICAL RESULTS

A. Estimation of the Recurrent Region

Fig. 2 shows $\mathcal{A}$ for the three-state finite-state Markov channel proposed by Suematsu and Imai (SI channel) for burst errors in magneto-optical recording. This channel has:

$$P = \begin{pmatrix} .999876 & 1.05358 \times 10^{-4} & 1.79010 \times 10^{-5} \\ .752429 & .247571 & 0 \\ 0.0458725 & 0 & .954127 \end{pmatrix}$$

and $Q = \left(2.31163 \times 10^{-4}, .4902, .4902\right)$ [14]. In this case, the log probabilities were rounded to the nearest integer in the computation of $H_k(\cdot)$; this approximation permits using the search technique to find the recurrent region without substantially modifying the state metric recursion. Note that for this channel, the recurrent region consists of disjoint regions.

For several channels of interest, Table I shows an upper bound on the recurrent region volume. Also shown are the maximum state metric differences $\Delta^{*,\mathrm{v}}$, found by searching the set $\mathcal{A}$. The four-state PR2 channel, $h(D) = (1+D)^2$ is of interest in optical recording, and the eight-state EPR4 channel, $h(D) = (1-D)(1+D)^2$, is of interest in magnetic recording.

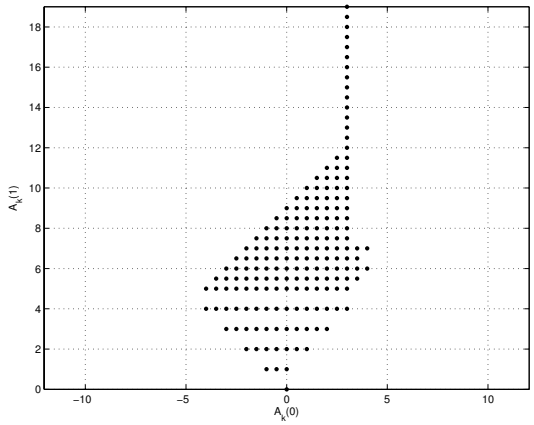

Fig. 2. Lattice points inside the recurrent region for the SI channel.

B. Gains for Lookup-Table Implementation

Although the nominal complexity of the lookup-table BCJR implementation is three table lookup operations per bit, the true complexity lies in the table size. An application of (3) is to estimate this table size. Assume that the received symbol y_k is quantized to n_y bits. With the conventional BCJR state metrics quantized to n_b bits per state, we find the size of the lookup table which gives the same state metric quantization error. The lookup table size for the forward recursion, for ISI channels (with zero *a priori* information) is:

$$2^{(n_b - n_b^{gain})\, M + n_y} \times \lceil (n_b - n_b^{gain})\, M \rceil \text{bits}.$$

The lookup table sizes with received symbol quantization $n_y = 5$ bits are shown in Table I. The comparison is made with the conventional implementation with state metric quantization with $n_b = 6$ bits/state. Also shown is the lookup table size that would be required for detecting the SI channel, assuming that the *a priori* information G_k is quantized with 5 bits, and the received information y_k is 0 or 1.

C. Application to Turbo Equalization

The application of the lookup-table BCJR implementation to LDPC-coded turbo equalization is considered, for the case of the Gilbert-Elliot channel with:

$$P = \begin{pmatrix} 1 - p_{01} & p_{01} \\ .06 & .94 \end{pmatrix}$$

and $Q = (0.01, 0.50)$. It was found by the search technique that the non-zero state metric $A_k(0)$ was bounded between $(-4, 3)$ and that $B_k(0)$ was bounded between $(-4, 8)$. Only uniform quantizers were considered for quantization of the state metrics. Fig. 3 shows the bit error rate performance with various spacing δ in the forward and backward recursion quantizer. As can be seen, as the quantizer codepoint spacing δ decreases, the performance rapidly approaches that of the turbo-equalized system using a floating-point BCJR implementation. Also shown is the performance of a non-turbo-equalized system, where the decoder assumes that the channel is a binary symmetric channel with the same error rate as the stationary error probability of the Gilbert-Elliot channel. As can be seen, turbo equalization imparts an advantage to such a system.

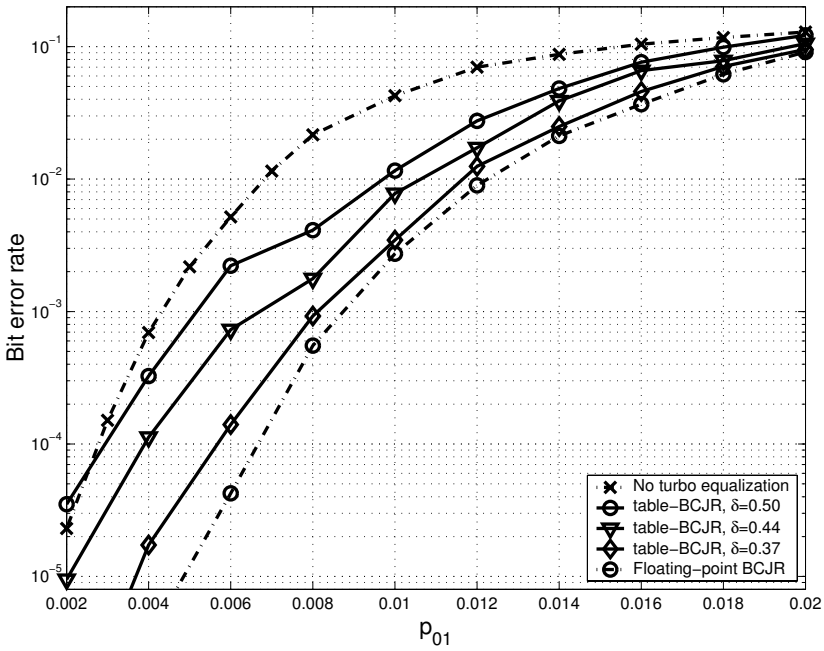

Fig. 3. Simulation results for a rate 1/2, block length 2640 LDPC code, with 10 iterations of turbo equalization, on the Gilbert-Elliot channel.

V. Conclusions

We have proposed a technique to find the recurrent region for the trellises of ISI channels and FSM channels. One application of this technique is to find benefits of vector quantization of state metrics. For the channels considered, this improvement was approximately 3 bits per state. This is also a contribution in understanding quantization and the vector nature of state metrics for the BCJR and Viterbi algorithms.

References

[1] T. Souvignier, A. Friedman, M. Öberg, P. H. Siegel, R. E. Swanson, and J. K. Wolf, "Turbo codes for PR4: Parallel versus serial concatenation," in *Proceedings IEEE International Conference on Communications*, (Vancouver, BC, Canada), pp. 1638–1642, IEEE, June 1999.

[2] J. L. Fan, A. Friedmann, E. Kurtas, and S. W. McLaughlin, "Low density parity check codes for magnetic recording," in *Proceedings 37th Annual Allerton Conference on Communication, Control, and Computing*, (Monticello, IL, USA), pp. 1314–1323, September 1999.

[3] T. Wadayama, "An iterative decoding algorithm of low density parity check codes for hidden Markov noise channels," in *International Symposium on Information Theory and its Applications*, (Honolulu, HI, USA), IEEE, November 2000.

[4] J. Garcia-Frias and J. D. Villasenor, "Turbo decoding of Gilbert-Elliot channels," *IEEE Transactions on Communications*, vol. 50, pp. 357–363, March 2002.

[5] J. Garcia-Frias, "Decoding of low-density parity check codes over finite-state binary markov channels," *IEEE Transactions on Communications*, vol. 52, pp. 1840–1843, November 2004.

[6] A. W. Eckford, F. R. Kschischang, and S. Pasupathy, "Designing good LDPC codes for Markov-modulated channels," in *Proceedings of IEEE International Symposium on Information Theory*, (Chicago, IL), IEEE, June 2004.

[7] R. Calderbank, P. Fishburn, and P. Siegel, "State-space characterization of Viterbi detector path metric differences," in *Twenty-Sixth Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, California), pp. 940–944, October 1992.

[8] B. Kurkoski, P. Siegel, and J. Wolf, "Soft-output detector for partial-response channels using vector quanitzation," in *IEEE Transactions on Magnetics*, IEEE, 2005. Accepted for publication.

[9] G. D. Forney, Jr., "The Viterbi algorithm," *Proceedings of the IEEE*, vol. 61, pp. 268–277, March 1973.

[10] P. Robertson and P. Hoeher, "Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding," *European Transactions on Telecommunications and Related Technologies*, vol. 8, pp. 119–125, March – April 1997.

[11] P. Siegel, C. B. Shung, T. D. Howell, and H. K. Thapar, "Exact bounds for Viterbi detector path metric differences," in *Proceedings of 1991 IEEE International Conference on Acoustics, Speech, and Signal Processing*, (Toronto, Canada), pp. 1093–1096, May 1991.

[12] E. Boutillon, W. J. Gross, and P. G. Gulak, "VLSI architectures for the MAP algorithm," *IEEE Transactions on Communications*, vol. 51, pp. 175–185, February 2003.

[13] A. P. Hekstra, "An alternative to metric rescaling in Viterbi decoders," *IEEE Transactions on Communications*, vol. 37, pp. 1220–1222, November 1989.

[14] T. Suematsu and H. Imai, "Estimation method for three state compound channel model," *Electronic Letters*, vol. 29, pp. 96–98, January 1993.