SEQUENTIAL DETECTION FOR SPARSE CHANNELS VIA A MULTIPLE TREE ALGORITHM

Weiwei Zhou, Jill K. Nelson

Department of Electrical and Computer Engineering, George Mason University

ABSTRACT

In this paper, we propose a tree-search based approach to detecting symbols transmitted over a sparse intersymbol interference channel. The proposed method uses a novel multiple-tree structure that exploits the channel sparsity to reduce computational complexity. By using parallel tree searches to perform data detection, the algorithm avoids the redundant likelihood computations introduced by inactive taps in the sparse channel impulse response. Simulation results show that, for moderate to high SNR, the multiple tree-search algorithm can reduce complexity by a factor of approximately 30 relative to the conventional Viterbi algorithm and by a factor of nearly 4 relative to multi-trellis methods.

Index Terms— Sequential detection, sparse channels, tree search, Viterbi algorithm, multi-trellis.

1. INTRODUCTION

Sparse intersymbol interference (ISI) channels are encountered in a wide range of applications, such as underwater acoustic communications [1] and high definition television (HDTV) systems [2]. The channels in these applications have a very large memory but only a small number of non-zero taps. The large memory of these channels often renders conventional maximum likelihood (ML) detection approaches, such as the Viterbi Algorithm (VA), prohibitively complex.

Numerous methods have been proposed to reduce the computational burden of data detection be exploiting channel sparsity. In [3], it was shown that only a small number of the filter coefficients in the decision feedback equalizer (DFE) are significant for sparse ISI channels, and a method for predicting the significant taps was proposed. Algorithms to find the locations of such coefficients were well studied in [4, 5]. A parallel trellis Viterbi algorithm (PTVA) was proposed in [6], but it is applicable only to zero padding channels, i.e., channels that have an equal number of zeros between active (non-zero) channel taps. In [7], a multi-trellis Viterbi algorithm (MVA) was designed for general sparse channels using multiple irregular trellises. It was observed in [6] and [7] that when using the VA for a sparse channel, many paths in the trellis have the same probability metric due to the large number of non-active channel taps. Both the PTVA and MVA were proposed to approximate the ML solution and reduce complexity by avoiding redundant path metric computations, but their complexity still grows exponentially with the number of active channel taps.

In this paper, we propose a computationally efficient tree-search based sequential detection method called the multiple tree algorithm (MTA). The proposed technique makes use of the stack algorithm (SA) for tree search, which was originally developed for decoding convolutional and tree codes [8]. Similar to the PTVA and MVA, the proposed algorithm reduces complexity by eliminating repeated metric computations within the detection process. However, it also takes advantage of the efficiency of the SA at moderate-to-high SNR to further reduce the computational load for long sparse channels with a large number of active taps. In contrast to the MVA, which uses a multi-trellis structure, the proposed method employs multiple trees, each with a reduced number of branches relative to a full search tree and each producing estimates of a subset of the transmitted bits. Each subtree takes a subset of the received sequence as input and obtains soft estimates of the bits that are not available to it from other subtrees. By eliminating paths with the same metric, the MTA significantly improves sequential detection efficiency for sparse ISI channels.

2. SYSTEM MODEL AND BACKGROUND

2.1. Sparse ISI channel model

We consider a discrete-time, baseband equivalent communication system. A length-N sequence of information symbols $\mathbf{X}_1^N = \{x_1, \ldots, x_k, \ldots, x_N\}$ is passed over a length-L_h sparse ISI channel $\mathbf{h}$ with additive white Gaussian noise (AWGN) $w_k \sim \mathcal{N}(0, \sigma^2)$. The channel output sequence $\mathbf{Y}_1^N = \{y_1, \ldots, y_k, \ldots, y_N\}$ is then processed by a detector to generate symbol estimates $\hat{\mathbf{X}}_1^N = \{\hat{x}_1, \ldots, \hat{x}_k, \ldots, \hat{x}_N\}$. We assume that the receiver knows the sparse channel, $\mathbf{h}$,

$$\mathbf{h} = [h_0, 0, \ldots, 0, h_1, 0, \ldots, 0, h_{L_a-1}]^T. \quad (1)$$

The channel has only L_a active (non-zero) taps, $\mathbf{h}_a = [h_0, h_1, \ldots, h_{L_a-1}]^T$, where $L_a \ll L_h$. Let p_i denote the position of the i-th active channel tap. The received signal at

time instant k can be expressed as

$$y_k = \sum_{i=0}^{L_a-1} h_i x_{k-p_i} + w_k = \mathbf{h}_a^T \mathbf{X}_{k-p_0}^{k-p_{L_a-1}} + w_k, \quad (2)$$

where $\mathbf{X}_{k-p_0}^{k-p_{L_a-1}} = \left[x_{k-p_0}\, x_{k-p_1}\, \ldots\, x_{k-p_{L_a-1}}\right]$ are the input bits on which y_k depends. For simplicity, we assume the input sequence $\mathbf{X}_1^N$ is BPSK modulated, i.e., $x_k \in \{+1, -1\}$. It is straightforward to extend the proposed algorithm to M-ary modulation schemes.

2.2. Stack algorithm for tree search

The SA is a best-first tree-search technique that extends the single most likely path at each iteration [8]. When the tree represents the space spanned by a sequence of transmitted bits, the SA can be used to approximate the ML solution to sequence detection. In such applications, each path in the tree represents a possible realization of the transmitted sequence. Each path has an associated metric, which expresses the likelihood that the corresponding bit sequence was transmitted, conditioned on the observations. A set of possible paths and their metrics are stored in a stack (or list) in order of decreasing metric value. At each time step, the SA extends the path with the highest likelihood, e.g. the top path in the stack. The algorithm terminates when the top path in the stack reaches a leaf of the tree, or equivalently when the top path represents a full block of transmitted bits.

When the SA is used to detect data transmitted over an ISI channel, the metric for a given path is governed by the symbol sequence associated with that path, the channel impulse response, and the output observations. When the ISI channel has many non-active taps, multiple paths in the tree will have the same metrics, resulting in a large number of redundant calculations. In Section 3, we propose a novel sequential detection method wherein we replace the single search tree with multiple parallel trees to avoid repeated metric computations.

3. STACK-BASED MULTIPLE TREE ALGORITHM

Rather than searching a single tree, the MTA uses multiple parallel subtrees designed to avoid redundant metric computations. For example, given a simple channel $\mathbf{h} = [h_0, 0, h_1]$, two subtrees can be constructed. One estimates the transmitted sequence $\{x_1, x_3, \ldots\}$, and the other estimates $\{x_2, x_4, \ldots\}$. The construction of the subtrees is determined by the positions of the active channel taps and is designed such that no two paths have the same metric.

Let J denote the number of parallel subtrees. The channel output sequence $\mathbf{Y}_1^N$ is divided into J subsets $\{\mathbf{Y}_j^{N_j}\}$, $j = 1, \ldots, J$, where $\mathbf{Y}_j^{N_j} = \{y_j, y_{j+J}, \ldots, y_{N-J+j}\}$ is the input to the j-th subtree. $\mathbf{X}_j^{N_j} = \{x_j, x_{j+J}, \ldots, x_{N-J+j}\}$ is the subset of the transmitted sequence estimated in the j-th subtree and is referred to as the state of subtree j. When the MTA is used, the channel output y_k may be fed to a subtree whose state does not include all the symbols x_{k-p_i}, $i = 0, \ldots, L_a - 1$, on which y_k is dependent. We obtain soft estimates of such symbols from the subtrees in which they are contained and incorporate the soft estimates into the path metric as needed. The path metric for the MTA is derived in Section 3.1, and the selection of the number of subtrees J is described in Section 3.2. A step-by-step description of the MTA is given in Section 3.3.

3.1. Derivation of the path metric of the MTA

The SA governs the order in which the paths of each subtree are explored. The sequence represented by each path is a non-contiguous subset of the transmitted sequence. As a result, the conventional sequential detection path metric must be modified to incorporate soft estimates of the symbols not contained in each subtree state. Let $\mathbf{X}_j^{n_j}$ denote the first $1 + (n_j - j)/J$ symbols estimated by the j-th subtree. In subtree j, the path metric expresses the likelihood that the information sequence $\mathbf{X}_j^{n_j}$ is transmitted given knowledge of the full received sequence $\mathbf{Y}_j^{N_j}$,

$$P(\mathbf{X}_j^{n_j} | \mathbf{Y}_j^{N_j}) = \frac{P(\mathbf{Y}_j^{N_j} | \mathbf{X}_j^{n_j}) P(\mathbf{X}_j^{n_j})}{P(\mathbf{Y}_j^{N_j})}. \quad (3)$$

In order to develop a closed-form expression for the path metric that can be computed with moderate complexity, we make two assumptions: 1) The received sequence beyond the path of interest, $\mathbf{Y}_{n_j+J}^{N_j}$, when conditioned on $\mathbf{X}_j^{n_j}$, is independent of the previous received sequence $\mathbf{Y}_j^{n_j}$. 2) The sequence $\mathbf{Y}_{n_j+J}^{N_j}$ is independent of the first $1 + (n_j - j)/J$ symbols $\mathbf{X}_j^{n_j}$ that subtree j estimates. Eliminating $P(\mathbf{Y}_j^{N_j})$ since it is equal for all paths, we can simplify the expression in (3) and write the path metric for the input sequence $\mathbf{X}_j^{n_j}$ as

$$\begin{aligned} m(\mathbf{X}_j^{n_j}) &= P(\mathbf{Y}_j^{N_j} | \mathbf{X}_j^{n_j}) P(\mathbf{X}_j^{n_j}) \quad (4) \\ &\approx P(\mathbf{X}_j^{n_j}) P(\mathbf{Y}_j^{n_j} | \mathbf{X}_j^{n_j}) P(\mathbf{Y}_{n_j+J}^{N_j} | \mathbf{X}_j^{n_j}) \\ &\approx P(\mathbf{X}_j^{n_j}) P(\mathbf{Y}_j^{n_j} | \mathbf{X}_j^{n_j}) P(\mathbf{Y}_{n_j+J}^{N_j}), \end{aligned}$$

where the second and third lines in (4) incorporate assumptions 1 and 2, respectively. Assuming the information symbols are independent and equally likely to be ± 1, the prior probability of the transmitted symbols is given by $P(\mathbf{X}_j^{n_j}) = (\frac{1}{2})^{1+(n_j-j)/J}$.

Since the SA extends only the path with the largest metric at each stage, the paths stored in the stack will generally be of varying lengths. $P(\mathbf{Y}_{n_j+J}^{N_j})$, which serves as a bias term that compensates for varying path length, can be obtained by averaging over all possible sequences $\mathbf{X}_{n_j+J}^{N_j}$. Because this computation is prohibitively complex to use in the path metric, we assume the future received symbols y_k, $k = n_j + J, \ldots, N_j$,

are independent of each other, and we approximate the bias term by a product of Gaussian terms as in [9],

$$P(\mathbf{Y}_{n_j+J}^{N_j}) \approx \prod_{k=n_j+J}^{N_j} P(y_k) \approx \prod_{k=n_j+J}^{N_j} \frac{1}{\sqrt{2\pi\sigma_0^2}} exp\left(-\frac{y_k^2}{2\sigma_0^2}\right), \tag{5}$$

where $\sigma_0^2 = \sigma^2 + \mathbf{h}^T\mathbf{h} = \sigma^2 + \mathbf{h}_a^T\mathbf{h}_a$.

The likelihood term $P(\mathbf{Y}_j^{n_j}|\mathbf{X}_j^{n_j})$ is given by the cascade of the conditional probabilities of each $y_k \in Y_j^{n_j}$,

$$P(\mathbf{Y}_j^{n_j}|\mathbf{X}_j^{n_j}) = \prod_{k=j:J:n_j} P(y_k|\mathbf{X}_{k-p_0}^{k-p_{L_a-1}}). \tag{6}$$

As mentioned, the channel output y_k may be fed to a subtree whose state does not include all the bits on which y_k is dependent. Let us denote by $\mathbf{X}_{(a)}^j = \{x_{k-p_i} \in \mathbf{X}_{k-p_0}^{k-p_{L_a-1}}\}$ and $\mathbf{X}_{(u)}^j = \{x_{k-p_i} \notin \mathbf{X}_{k-p_0}^{k-p_{L_a-1}}\}$, $i \in \{0, \ldots, L_a-1\}$, the F bits available and the D bits unavailable, respectively, in the j-th subtree state. The conditional likelihood of y_k can be written as

$$\begin{aligned} P(y_k|\mathbf{X}_j^{n_j}) &= P\left(y_k|\mathbf{X}_{(a)}^j\right) \qquad (7)\\ &= \sum_l P\left(y_k|\mathbf{X}_{(a)}^j, \mathbf{X}_{(u)}^j = \mathbf{q}_l\right) P(\mathbf{X}_{(u)}^j = \mathbf{q}_l), \end{aligned}$$

where the vector $\mathbf{q}_l = \{q_l^d\}$, $1 \le d \le D$, denotes the l-th realization of a length-D BPSK sequence. Given the sequence $\mathbf{X}_{k-p_0}^{k-p_{L_a-1}}$, the received element y_k is Gaussian, $y_k \sim \mathcal{N}(\mathbf{h}_a^T\mathbf{X}_{k-p_0}^{k-p_{L_a-1}}, \sigma^2)$. Therefore,

$$P\left(y_k|\mathbf{X}_{(a)}^j, \mathbf{X}_{(u)}^j = \mathbf{q}_l\right) = \tag{8}$$
$$\frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{(y_k - \mathbf{h}_a^T\{\mathbf{X}_{k-p_0}^{k-p_{L_a-1}}\}_{\mathbf{X}_{(u)}^j=\mathbf{q}_l})^2}{2\sigma^2}\right),$$

where $\{\mathbf{X}_{k-p_0}^{k-p_{L_a-1}}\}_{\mathbf{X}_{(u)}^j=\mathbf{q}_l}$ denotes the sequence $\mathbf{X}_{k-p_0}^{k-p_{L_a-1}}$ with $\mathbf{X}_{(u)}^j = \mathbf{q}_l$. Noting that the elements of $\mathbf{X}_{(u)}^j$ are independent, we have

$$P(\mathbf{X}_{(u)}^j = \mathbf{q}_l) = \prod_{d=1}^{D} P(\mathbf{X}_{(u)}^j(d) = \mathbf{q}_l(d)), \tag{9}$$

where $P(\mathbf{X}_{(u)}^j(d) = \mathbf{q}_l(d))$ is the soft information that must be obtained from another subtree. Suppose, for example, that symbol $\mathbf{X}_{(u)}^j(d)$ is available in subtree t, $t \neq j$. If all paths in subtree t are explored to depth N_t, $P(\mathbf{X}_{(u)}^j(d) = \mathbf{q}_l(d))$ can be exactly computed based on the likelihoods of the paths for which $\mathbf{X}_{(u)}^j(d) = \mathbf{q}_l(d)$, i.e.,

$$P(\mathbf{X}_{(u)}^j(d) = \mathbf{q}_l(d)) = \sum_{\mathbf{X}_t^{N_t}:\, \mathbf{X}_j^{(u)}(d)=\mathbf{q}_l(d)} P(\mathbf{X}_t^{N_t}|\mathbf{Y}_t^{N_t}). \tag{10}$$

When the SA is used, however, not all paths in a subtree are explored. Thus, $P(\mathbf{X}_{(u)}^j(d) = \mathbf{q}_l(d))$ can be obtained only approximately based on the paths present in the stack for subtree t,

$$P(\mathbf{X}_{(u)}^j(d) = \mathbf{q}_l(d)) \approx \sum_{\mathbf{X}_t^{n_t} \in S_t:\, \mathbf{X}_j^{(u)}(d)=\mathbf{q}_l(d)} P(\mathbf{X}_t^{n_t}|\mathbf{Y}_t^{N_t}), \tag{11}$$

where S_t denotes the stack for subtree t.

The observation likelihood term, $P(\mathbf{Y}_j^{n_j}|\mathbf{X}_j^{n_j})$, can be obtained by substituting (7), which incorporates (8) and (9), into (6). Substituting (5) and (6) into (4) produces a closed-form expression for the path metric that is used in each subtree of the MTA.

3.2. Determination of the number of subtrees J

In order to determine the number of subtrees J, we use the criterion of minimizing the number of soft estimates exchanged among subtrees. Recall the expression of the sparse channel output given in (2). The first $p_1 - p_0$ channel outputs y_k, $k = 1, \ldots, p_1 - p_0$, are dependent only on single symbols x_k, $k = 1, \ldots, p_1 - p_0$, respectively. Outputs y_k, $k = p_1 - p_0 + 1, \ldots, N$, are dependent on multiple input symbols. In order to minimize the number of soft estimates exchanged, we need to maximize the number of symbols x_{k-p_i}, $i = 1, \ldots, L_a - 1$, available in subtree j. Due to the varying number of zeros between active channel taps, we can only guarantee that two symbols, x_k and x_{k-p_i}, are in subtree j. The first $p_1 - p_0$ symbols can be arranged to form the first bit for each subtree state, and x_k, $p_1 - p_0 < k < p_2$, can then be estimated without obtaining information from other subtrees; this is accomplished by setting $J = p_i - p_0$, $i = 1, \ldots, L_a - 1$. To estimate x_k, $k > p_1 - p_0$, subtree j must obtain $L_a - 2$ soft estimates from other trees.

3.3. Summary of the MTA

The MTA procedure is summarized below. Without loss of generality, we assume the sparse channel has $L_a = 3$ active taps.

1 Initialization: Construct $J = p_1 - p_0$ parallel subtree modules, each of which represents the search space spanned by transmitted symbols $\mathbf{X}_j^{N_j} = \{x_{Jl+j}\}$, where $j = 1, \ldots, J$ and $l = 0, 1, \ldots, (N/J - 1)$. Divide the received signal into J subsets, $\mathbf{Y}_j^{N_j} = \{y_{Jl+j}\}$. Insert a root node with metric 0 into a stack for each subtree. Set $j = 1$ and $l = 1$.

2 Extend the path at the top of S_j, the stack for subtree j, to all of its children and compute the new path metrics. The soft estimates generated by subtree $(J(l-1)+j-(p_2-p_1)) \oplus J$ are taken as input, where $\oplus$ denotes the modulo operation. For a branch at depth l, the metric is

computed using the estimate $P(x_{J(l-1)+j-(p_2-p_1)} = \pm 1)$ from subtree $(J(l-1)+j-(p_2-p_1)) \oplus J$ if $Jl + j \geq p_2$. Otherwise, no soft estimate needs to be obtained.

3 Store all explored paths and their metrics in S_j in order of decreasing metric value.

4 If the depth of the top path in S_j is l, go to **5**. Otherwise, return to **2**.

5 Within subtree j, generate soft estimate $P(x_{J(l-1)+j} = \pm 1)$ using (11); pass the soft estimate to subtree $(J(l-1)+j-(p_2-p_1)-1) \oplus J$ if $Jl + j \geq p_2$.

6 If $j < J$, set $j = j + 1$ and return to **2**. Otherwise, go to **7**.

7 If the top paths of all J stacks have reached a leaf of the corresponding subtree, i.e., $l = (N-J)/J$, stop and output the sequences from all J subtrees in order. Otherwise, set $l = l + 1$ and $j = 1$, and return to **2**.

4. COMPLEXITY ANALYSIS AND SIMULATIONS

To illustrate the complexity and performance of the proposed MTA, we compare it to the VA, SA and MVA for a sparse channel with length L_h and L_a active channel taps. We take the number of computations of a path metric as the complexity measure. For an input sequence with length N, the computational complexity of the conventional VA is NM^{L_h} [10], and the complexity of MVA is NM^{L_a} [7]. There is no closed-form expression for the complexity of MTA, as the number of path extensions for the SA varies with the received sequence. Only an upper bound on the complexity can be found [11]. Therefore, we evaluate the complexity of proposed MTA algorithm empirically for comparison with competing algorithms.

Simulations have been conducted for a length $L_h = 6$ sparse channel with $L_a = 3$ active taps, $\mathbf{h} = [h_0, 0, 0, h_1, 0, h_2]$. 1000 blocks of 1200 BPSK symbols are passed over the sparse channel. For each data block, the active channel taps $\mathbf{h}_a = [h_0, h_1, h_2]$ are drawn from a uniform distribution on interval $(0, 1)$, and the channel energy is normalized to 1. The stack size of the SA is set to 10^6 to make erasures rare. The average number of computations performed per transmitted block as a function of SNR is shown in Table 1 for the four detection methods considered. As SNR increases, the complexity of the VA and MVA remain constant, while the complexity of the MTA decreases dramatically, as would be expected of a stack-based algorithm. For SNR from 5 dB to 10 dB, the MTA achieves significant computational complexity reduction compared to the other three methods. For SNR greater than 10 dB, the SA and MTA are likely to follow a single path and explore very little of the tree, causing their similar complexity.

A performance comparison of the four methods is shown in Fig.1. The MTA suffers some performance loss compared to the VA, but its performance is very close to that of the SA and MVA. The performance difference among the methods decreases as SNR increases. Therefore, with only slight performance degradation, the proposed MTA provides a computationally efficient approach to data detection for sparse ISI channels at moderate SNRs.

Number of Computations				
	VA	**MVA**	**SA**	**MTA**
4dB	76800	9600	154518	74698
5dB	76800	9600	14429	5687
6dB	76800	9600	7551	4758
7dB	76800	9600	5059	3207
8dB	76800	9600	3286	2426
9dB	76800	9600	2726	2411
10dB	76800	9600	2512	2405
11dB	76800	9600	2408	2401

Table 1: Computations performed for the VA, MVA, SA and MTA using a length-1200 sequence for a length-$L_h = 6$ sparse channel with $L_a = 3$ active taps.

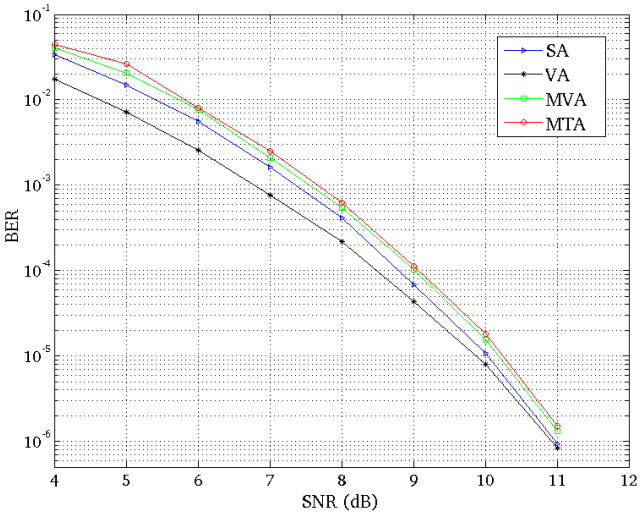

Fig. 1: Performance comparison of the VA, SA, MVA and MTA for a length-$L_h = 6$ sparse channel with $L_a = 3$ active taps.

5. CONCLUSIONS AND FUTURE DIRECTIONS

We have introduced a computationally efficient algorithm for sequential detection of data transmitted over a sparse ISI channel. By constructing and searching multiple trees, each of which is used to estimate a subset of the transmitted symbols, the proposed MTA reduces detection complexity by eliminating redundant computations of path likelihood metrics. Simulation results show that the MTA can achieve significant complexity reduction relative to competing trellis-based schemes for moderate to high SNR. Future work will focus on analyzing the quality of the soft information exchanged among subtrees and extending the MTA for application to time-varying channels.

6. REFERENCES

[1] C. R. Berger, Zhaohui Wang, Jianzhong Huang, and Shengli Zhou, "Application of compressive sensing to sparse channel estimation," *IEEE Communications Magazine*, vol. 48, no. 11, pp. 164 –174, November 2010.

[2] Lingyan Fan, Chen He, Dongjian Wang, and Lingge Jiang, "Efficient robust adaptive decision feedback equalizer for large delay sparse channel," *IEEE Transactions on Consumer Electronics*, vol. 51, no. 2, pp. 449 – 456, May 2005.

[3] Shaohua Chen, Jinglin Xiang, and Jie Shi, "Modified decision feedback equalizer (DFE) for sparse channels in underwater acoustic communications," *Proceedings of the 2006 IET International Conference on Wireless, Mobile and Multimedia Networks*, November 2006.

[4] M.J. Lopez and A.C. Singer, "A DFE coefficient placement algorithm for sparse reverberant channels," *IEEE Transactions on Communications*, vol. 49, no. 8, pp. 1334 –1338, August 2001.

[5] A.A. Rontogiannis and K. Berberidis, "Efficient decision feedback equalization for sparse wireless channels," *IEEE Transactions on Wireless Communications*, vol. 2, no. 3, pp. 570 – 581, May 2003.

[6] N.C. McGinty, R.A. Kennedy, and P. Hocher, "Parallel trellis Viterbi algorithm for sparse channels," *IEEE Communications Letters*, vol. 2, no. 5, pp. 143 –145, May 1998.

[7] N. Benvenuto and R. Marchesani, "The Viterbi algorithm for sparse channels," *IEEE Transactions on Communications*, vol. 44, no. 3, pp. 287 –289, March 1996.

[8] R. Johannesson and K. Zigangirov, *Fundamentals of Convolutional Coding*, IEEE Press, New York, 1999.

[9] J.K. Nelson and A.C. Singer, "Bayesian ML sequence detection for ISI channels," *Proceedings of the 40th Annual Conference on Information Sciences and Systems*, pp. 693 –698, March 2006.

[10] John G. Proakis and Masoud Salehi, *Digital Communications*, McGraw-Hill, 2007.

[11] F. Xiong, Q. Dai, and E. Shwedyk, "Computational complexity of sequential sequence estimation for intersymbol interference channels," *IEEE Transactions on Communications*, vol. 41, no. 2, pp. 332 –337, February 1993.