Fast Joint Source-Channel Decoding of Convolutional Coded Markov Sequences with Monge Property

Sorina Dumitrescu

Abstract

This work addresses the problem of joint source-channel decoding of a Markov sequence which is first encoded by a source code, then encoded by a convolutional code, and sent through a noisy memoryless channel. It is shown that for Markov sources satisfying the so-called Monge property, both the maximum a posteriori probability (MAP) sequence decoding, as well as the soft output Max-Log-MAP decoding can be accelerated by a factor of K without compromising the optimality, where K is the size of the Markov source alphabet. The key to achieve a higher decoding speed is a convenient organization of computations at the decoder combined with a fast matrix search technique enabled by the Monge property. The same decrease in complexity follows, as a by-product of the development, for the soft output Max-Log-MAP joint source channel decoding in the case when the convolutional coder is absent, result which was not known previously.

Key words: Joint source-channel decoding, maximum a posteriori probability sequence estimation, Max-Log-MAP algorithm, Markov sequence, Monge property.

The author is with the ECE Department at McMaster University, Canada. The author's e-mail address is sorina@mail.ece.mcmaster.ca. This work was presented in part at the 2007 IEEE Information Theory Workshop, Lake Tahoe, Sept. 2007.

I. INTRODUCTION

In traditional communications systems over noisy channels the source coder and channel coder are designed independently. This separation is motivated by Shannon's source-channel coding theorem which states that optimum performance can be achieved asymptotically in the code block length and complexity, by following this approach. However, in practice, the system's design may be constrained by delay and/or complexity. Hence, the separate approach could lead to severe degradation in performance.

Joint source-channel (JSC) decoding is a way to alleviate this problem, which has received considerable attention lately. The main idea is that suboptimal design of the source coder fails to remove completely the redundancy of the source. Thus, some residual redundancy [16] will be present at the input of the channel coder. This redundancy can be used solely or in addition to the redundancy provided by the channel coder to correct transmission errors. One way to take into account the redundancy left after a suboptimal quantization process is by modeling its output as a discrete Markov source. Many researchers have assumed such a model and investigated the JSC decoding of a discrete Markov source, which is applied a symbol by symbol source coder, followed or not by a channel coder, and is sent through a noisy channel. A natural way for JSC decoding in such systems is by maximum a posteriori probability (MAP) Markov sequence estimation. In [14], [2] MAP Markov sequence decoders were proposed for the case when the source codewords have fixed-length and no channel code is applied. In [7], [13], [17], [18] the case of variable-length source code was treated. MAP Markov sequence decoders for systems where the fixed-length or variable-length source coder is followed by a convolutional coder were considered in [12], [6], [8]. Soft output decoders of Markov sequences as components of iterative JSC decoding schemes have also been investigated [9], [10], [11].

This paper is mainly concerned with JSC decoding of a Markov sequence (MS), which is first compressed by a source coder (SC), then protected by a convolutional coder (CC), and sent through a memoryless noisy channel. The problem has been dealt with previously in [12], [10], [6], [8]. As noted in the previous work the cascaded chain formed by the MS followed by the SC and the CC (MS$\rightarrow$SC$\rightarrow$CC) can be modelled as a combined finite-state machine (CFSM), whose states are all the triples of the states of the three elements in the chain. Then, if the size

I of the Markov sequence is not known to the decoder, both the MAP sequence JSC estimation and the bit-level soft output JSC decoding can be performed on the bit-level trellis generated by the CFSM in $O(TK^2N)$ time, where K is the number of source symbols, N is the number of states of the CC, and T is the length in bits of the information sequence output by the SC. [10]. The authors of [12], [6] suggest that the number of states of the CFSM can be reduced thus decreasing the JSC decoding time. In [8] the case of a feedforward convolutional encoder is considered. The number of states in the reduced CFSM is quantified to $O(K^2 + N \log N)$ as opposed to $O(K^2N)$ in the full-size CFSM, which implies that the JSC decoding algorithm has lower time complexity than the one evaluated in [10] for general CC. Moreover, the authors of [8] show that the MAP sequence JSC decoding algorithm can be further accelerated without compromising the optimality, if $K > N$ and if the source satisfies the so-called Monge property.

In this work we address the case of a general convolutional encoder. In the general case a reduction of the state space of the CFSM is not always possible, hence the results of [8] do not hold. Therefore, it is desirable to find other means to expedite the decoding. We present in this work a way to speed up the JSC decoding if the Markov source satisfies the Monge property, i.e., if

$$\begin{aligned} &\log_2 P(a_{i'_2}|a_{i_1}) + \log_2 P(a_{i'_1}|a_{i_2}) \leq \\ &\log_2 P(a_{i'_1}|a_{i_1}) + \log_2 P(a_{i'_2}|a_{i_2}), \\ &\text{for all } 1 \leq i_1 < i_2 \leq K, 1 \leq i'_1 < i'_2 \leq K. \end{aligned} \tag{1}$$

It was shown in [18] that if the discrete Markov source is the output of a scalar quantizer applied to a continuous source[1] whose joint probability density function (pdf) $f(u, v)$ of two consecutive samples satisfies

$$\begin{aligned} &\log_2 f(v', u) + \log_2 f(v, u') \leq \\ &\log_2 f(v', u') + \log_2 f(v, u), \end{aligned} \tag{2}$$

for any real values $u < u'$ and $v < v'$, then the condition (1) holds. Moreover, as proved in [5], if the second partial derivative $\partial^2(\log f)/\partial u \partial v$ exists, then inequality (2) is valid if and

[1]Although a quantized Markov chain may not necessarily be a Markov chain, here we assume this to be a good approximation.

only if $\partial^2(\log f)/\partial u \partial v \geq 0$ for all real values u, v. The last relation is clearly true when the joint pdf $f(\cdot,\cdot)$ is Gaussian. Consequently, the Monge property (1) holds for a scalar quantized Gaussian-Markov source.

In this work we show how the Monge property, when satisfied by the Markov source, can be taken advantage of to accelerate by a factor of K the JSC MAP sequence decoding. This complexity reduction result holds for both cases when the size of the input Markov sequence is or is not known to the decoder. Our strategy for increased speed is based on a convenient organization of the computations combined with the use the fast matrix search algorithm of [1] for finding all column maxima in a totally monotone matrix.

We mention that this efficient matrix search was also applied in [18] to accelerate the MAP sequence decoding for Markov sources with the Monge property. However, in [18] no channel coder was present. Incorporating a convolutional coder in the scheme, as in the present work, introduces new challenges in applying this technique, whose solution is not straightforward. In [8] these challenges were overcome in the case of feedforward convolutional encoder, based on the particularities of such an encoder, without providing an obvious extension to the case of general CC. In this work we clarify how the Monge property can be exploited to reduce the MAP sequence JSC decoding complexity in the setting of a general CC.

Furthermore, we show that the complexity reduction by a factor of K can be achieved for soft output Max-Log-MAP JSC decoding as well. Max-Log-MAP algorithms are suboptimal, but more efficient algorithms for symbol or bit a posteriori probabilities (APPs) computation [15], where the processing is performed exclusively in the logarithmic domain. Our complexity reduction result for soft output Max-Log-MAP JSC decoding is new for the case of feedforward convolutional encoder as well. Moreover, this result can be trivially extended to the case when the CC is absent, providing an additional contribution over [18].

The paper is organized as follows. In the next section we present the setting considered in this work and formulate the problem of JSC decoding by MAP sequence estimation. Section 4 introduces the weighted directed acyclic graph which will be used for JSC decoding. It is shown that the problem of MAP sequence estimation is equivalent to the maximum-weight path problem in this graph. Section 5 presents a convenient organization of the computations needed to solve

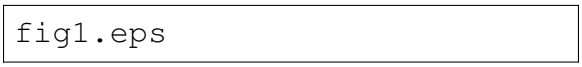

Fig. 1. The sequential operations of the system.

this problem, which will be further useful toward achieving the speed up in the case of Markov sources with the Monge property. Section 6 shows how the Monge property can be exploited to accelerate by a factor of K the MAP sequence JSC decoding. Furthermore, in Section 7 we demonstrate that an analogous technique can be used to achieve the same complexity reduction of bit-level soft output Max-Log-MAP JSC decoding.

II. Problem Formulation

We consider a first order Markov source (MS) over an alphabet of K symbols $\{a_1, a_2, \cdots, a_K\}$, with conditional probabilities $P(a_k|a_j)$, $1 \leq k, j \leq K$, and initial probabilities $P(a_k)$, $1 \leq k \leq K$. Let $\mathbf{u} = u_1 u_2 \cdots u_I$ denote the sequence generated by the MS. The Markov sequence $\mathbf{u}$ is encoded by a symbol-by-symbol source coder (SC), which can be a fixed or a variable-length coder, into the bitstream $\mathbf{v} = v_1 v_2 \cdots v_T$, $v_t \in \{0, 1\}$, $1 \leq t \leq T$. Let c_k denote the binary codeword assigned to the source symbol a_k, $1 \leq k \leq K$.

Further a convolutional encoder (CC) is applied to the sequence $\mathbf{v}$, generating a new bitstream $\mathbf{x} = x_1 x_2 \cdots x_M$, $x_m \in \{0, 1\}$, $1 \leq m \leq M$. We assume that the CC has rate $1/\beta$, and memory order μ. The CC itself can be described by a state transition diagram. We denote the states by $S_1, \cdots, S_N$, where $N = 2^\mu$ and S_1 is the initial state. For each state S_n and each bit $b \in \{0, 1\}$, there is a transition with input bit b and some output sequence of β bits $b_1 b_2 \cdots b_\beta$, from the state S_n to some state $S_{n'}$.

Finally, the bitstream $\mathbf{x}$ is sent through a noisy memoryless channel. The bit sequence received at the other end is denoted by $\mathbf{y} = y_1 y_2 \cdots y_M$. The block diagram of the whole process is shown in Figure 1.

The problem of joint source-channel (JSC) maximum a posteriori probability (MAP) sequence decoding is, given the sequence $\mathbf{y} = y_1 y_2 \cdots y_M$ output by the noisy channel, to find the input Markov sequence $\mathbf{u} = u_1 u_2 \cdots u_I$ of maximal a posteriori probability $P(\mathbf{u}|\mathbf{y})$.

By Bayes' Theorem we have $P(\mathbf{u}|\mathbf{y}) = P(\mathbf{u})P(\mathbf{y}|\mathbf{u})/P(\mathbf{y})$. Thus, maximizing $P(\mathbf{u}|\mathbf{y})$ is equivalent to maximizing $P(\mathbf{u})P(\mathbf{y}|\mathbf{u})$, since $\mathbf{y}$ is fixed. On the other hand, the bitstream $\mathbf{x}$ is completely determined by the Markov sequence $\mathbf{u}$ and vice versa, thus $P(\mathbf{y}|\mathbf{u})$ equals the channel transition probability $P_{ch}(\mathbf{y}|\mathbf{x})$. Since the channel is memoryless, it follows that $P_{ch}(\mathbf{y}|\mathbf{x}) = \Pi_{m=1}^{M} P_{ch}(y_m|x_m)$. Moreover, because $\mathbf{u}$ is generated by a first order MS, we obtain that $P(\mathbf{u}) = P(u_1)\Pi_{i=2}^{I} P(u_i|u_{i-1})$. Thus, the problem of MAP sequence JSC decoding becomes equivalent to finding the Markov sequence $\mathbf{u} = u_1 u_2 \cdots u_I$ and/or the corresponding bitstream $\mathbf{x} = x_1 x_2 \cdots x_M$ such that the product

$$P(u_1)\Pi_{i=2}^{I} P(u_i|u_{i-1})\Pi_{m=1}^{M} P_{ch}(y_m|x_m) \tag{3}$$

is maximized, given the bitstream $\mathbf{y} = y_1 y_2 \cdots y_M$ output by the noisy channel. Note that the number of source symbols I cannot be inferred from M in the case of a variable-length SC. We assume for now that I is not made available to the decoder by any other means. Therefore, the maximization of (3) has to be performed over all sequences $\mathbf{u}$ which generate a bitstream $\mathbf{v}$ of length T. Further, notice that maximizing (3) is equivalent to maximizing its base 2 logarithm, i.e.,

$$\log_2 P(u_1) + \sum_{i=2}^{I} \log_2 P(u_i|u_{i-1}) + \sum_{m=1}^{M} \log_2 P_{ch}(y_m|x_m) \tag{4}$$

As noted in [12], [10], [6], the generation of the bitstream $\mathbf{x}$, which is produced by the composite action of MS, SC and CC, can be modeled by a stochastic finite-state machine, called the combined finite-state machine (CFSM) in [8], whose states are all the triples (c_k, L_j, S_n), $1 \leq k \leq K$, $1 \leq j \leq K-1$, $1 \leq n \leq N$, where $L_1, L_2, \cdots, L_{K-1}$ denote the internal nodes of the binary tree describing the source code.

Therefore, as observed in [12], [10], [6], the JSC MAP sequence decoding can be performed by running the Viterbi algorithm on the bit-level trellis obtained by expanding in time the state diagram of the CFSM. Since the total number of states of the CFSM is $K(K-1)N$ (omitting those states which can be reached only once in a sequence of transitions), this leads to an $O(TK^2N)$ time decoding algorithm. In [12], [6] it was suggested that the state-space size of the CFSM can be reduced by eliminating those states unreachable from the initial one, thus decreasing the decoding time. In [8] the state-space size of the reduced CFSM was evaluated to $O(K^2 + N\log N)$ for the case of feedforward convolutional encoder, result which establishes to

$O(TK^2 + TN \log N)$ the time complexity of the Viterbi decoding on the reduced CFSM-based trellis.

However in the setting of a general convolutional coder such a reduction of the CFSM is not always possible, consequently other means to speed up the JSC decoding are desirable. The contribution of this work is a strategy to accelerate the JSC decoding for Markov sources satisfying the Monge property (1). To proceed toward our goal we need to introduce a different weighted directed acyclic graph (WDAG) G for the joint source-channel decoding. This is accomplished in the following section. Note that for the development in the next two sections to hold, the Markov source does not need to satisfy the Monge property.

III. Decoder Graph

In this section we describe the WDAG G used for decoding. This graph can be regarded as a modification of the bit-level trellis. While every branch in the trellis corresponds to decoding a single bit, in order to obtain this graph we contract every path corresponding to decoding of a source codeword into a single edge. Intermediate nodes on such a path are not needed anymore and are therefore eliminated. As a result, our graph has roughly K times less nodes than the trellis, this reduction leading to a decrease in decoding space complexity. We mention that the idea of aggregating bit-level branches into codeword-level branches has been used in the prior literature, for example in [13], [18], for the JSC decoding of variable-length coded Markov sequences when the CC is absent.

We also point out that the decoder graph considered here differs from the graph used in [8] for efficient JSC decoding of Markov sources with Monge property, in the case of a feedforward convolutional encoder. The graph of [8] is also obtained starting from the bit-level trellis by contracting some paths into edges, but only a subset of the paths corresponding to decoding a source codeword of minimum size μ (the memory size of the CC). Consequently, the decoder graph in the present work is less complex.

The WDAG G contains $T + 1$ layers labelled from 0 to T, where T is the length of the bit sequence $\mathbf{v}$ output by the SC, hence $T = M/\beta$. Each layer $t, t \geq 1$, corresponds to the t-th bit of this sequence. The basic idea is to have for each pair (c_k, S_n) a corresponding node on each layer. Such a node corresponds to the state (c_k, L_1, S_n) in the CFSM, where L_1 denotes the root

Fig. 2. Decoder graph G: all transitions starting at layer t and corresponding to source codeword 0.

of the source code tree. In order to take into account the cases when the CFSM can be reduced as well, we consider only pairs (c_k, S_n) such that (c_k, L_1, S_n) is reachable from the initial state in the CFSM, and denote by $\mathcal{C}$ the set of all these pairs. The nodes on layer t of the graph G are denoted by (c_k, S_n, t) for all $1 \leq t \leq T$, $1 \leq k \leq K$ and $1 \leq n \leq N$ such that $(c_k, S_n) \in \mathcal{C}$. Layer 0 contains only a single node, $(c_0, S_1, 0)$, which is the source node of the graph. The final nodes of the graph are all nodes on layer T.

The edges of the graph are all ordered pairs $((c_k, S_n, t), (c_{k'}, S_{n'}, t + |c_{k'}|))$ such that there is a sequence of transitions in the state diagram of the CC from the state S_n to the state $S_{n'}$, with sequence of input bits $c_{k'}$ and some sequence of output bits $b_1 b_2 \cdots b_{\beta|c_{k'}|}$. The weight assigned to such an edge is defined as

$$\log_2 P_{ch}(y_{t\beta+1} \cdots y_{t\beta+\beta|c_{k'}|} | b_1 b_2 \cdots b_{\beta|c_{k'}|}) + \log_2 P(a_{k'}|a_k). \quad (5)$$

Figures 2 and 3 illustrate the transitions starting at some layer t in the decoder graph G

Fig. 3. Decoder graph G: all transitions starting at layer t and corresponding to source codewords 10 or 11.

corresponding to Example 1. To avoid the picture agglomeration we have broken this set of transitions into two: those corresponding to the codeword 0 (depicted in Figure 2 and those corresponding to codewords 10 and 11 illustrated in Figure 3.

Example 1. Consider a Markov source over a three-symbol alphabet, the SC with codewords $c_1 = 0$, $c_2 = 10$, $c_3 = 11$, and a rate $1/2$ recursive systematic CC with memory order $\mu = 2$. Table 1 shows the transition matrix of the CC, where the states are denoted as follows: $S_1 = 00$, $S_2 = 01$, $S_3 = 10$, $S_4 = 11$.

It is easy to see that any path in the graph G from the source (the node at layer 0) to a final node can be mapped uniquely to a sequence of Markov source symbols $\mathbf{u} = u_1 u_2 \cdots u_I$ which results in a T-length bit sequence after the SC is applied. Moreover, the weight of the path equals the quantity (4). Therefore the problem of JSC MAP sequence decoding is equivalent to finding the maximum-weight path from the source to the final node in G. The number of nodes of this graph is clearly $O(T|\mathcal{C}|)$, and the number of edges is $O(TK|\mathcal{C}|)$ since there are K edges starting from each node. Consequently, the maximum-weight path problem can be

	S_1	S_2	S_3	S_4
S_1	$0/00$		$1/11$	
S_2	$1/11$		$0/00$	
S_3		$1/10$		$0/01$
S_4		$0/01$		$1/10$

TABLE I

TRANSITION MATRIX OF THE CC OF EXAMPLE 1. THE ENTRIES $b/b'_1b'_2$ MARK VALID TRANSITIONS WITH INPUT b AND OUTPUT $b'_1b'_2$.

solved in $O(TK|\mathcal{C}|)$ time and $O(T|\mathcal{C}|)$ space. If the CFSM cannot be reduced then $|\mathcal{C}| = KN$, thus leading to the same decoding time complexity as for the bit-level trellis, i.e., $O(TK^2N)$. However, the graph G has fewer nodes, only $O(TKN)$ versus $O(TK^2N)$ of the trellis, thus yielding a space complexity reduced by a factor of K.

In order to achieve the time complexity reduction for Markov sources with the Monge property we have to further organize the computations for finding the maximum-weight path in a convenient way. This is the topic of the next section.

IV. NEW ORGANIZATION OF COMPUTATIONS

Before exposing our strategy we need to introduce some notations. For this note that, given n and k', there is a unique n' such that there is a sequence of transitions in the CC diagram from S_n to $S_{n'}$ with input $c_{k'}$. We denote this value of n' by $\alpha(n, k')$. On the other hand, given k' and n', there may be more values n such that there is a sequence of transitions from S_n to $S_{n'}$ with input $c_{k'}$. We denote by $\alpha^{-1}(n', k')$ the set of these values of n. Further, note that the first term in (5) does not depend on k, and since n' is completely determined by n and k', it follows that this term is completely determined by t, n and k'. Therefore we will denote it by $\omega(t, n, k')$, i.e.,

$$\omega(t, n, k') \triangleq$$

$$\log_2 P_{ch}(y_{t\beta+1} \cdots y_{t\beta+\beta|c_{k'}|} | b_1 b_2 \cdots b_{\beta|c_{k'}|}). \tag{6}$$

For any node ν of G we denote by $W(\nu)$ the weight of the maximum-weight path from the source to that node. Notice that any edge ending in some node $(c_{k'}, S_{n'}, t+|c_{k'}|)$ has to start from some node on layer t, more specifically, it can start from any node (c_k, S_n, t) such that $n \in \alpha^{-1}(n', k')$ and $(c_k, S_n) \in \mathcal{C}$. Therefore the following relation holds

$$W(c_{k'}, S_{n'}, t+|c_{k'}|) = \max_{n \in \alpha^{-1}(n',k'),(c_k,S_n)\in\mathcal{C}} \{W(c_k, S_n, t) + \omega(t, n, k') + \log_2 P(a_{k'}|a_k)\}.$$

Because the term $\omega(t, n, k')$ does not depend on c_k we further obtain the following equality

$$W(c_{k'}, S_{n'}, t+|c_{k'}|) = \max_{n \in \alpha^{-1}(n',k')} \{\omega(t, n, k') + \max_{k,(c_k,S_n)\in\mathcal{C}} (W(c_k, S_n, t) + \log_2 P(a_{k'}|a_k))\}. \quad (7)$$

Denote now for all k', n' and $n \in \alpha^{-1}(n', k')$,

$$W_n(c_{k'}, S_{n'}, t+|c_{k'}|) \triangleq \max_{k,(c_k,S_n)\in\mathcal{C}} (W(c_k, S_n, t) + \log_2 P(a_{k'}|a_k)). \quad (8)$$

Then relation (7) is equivalent to

$$W(c_{k'}, S_{n'}, t+|c_{k'}|) = \max_{n \in \alpha^{-1}(n',k')} \{\omega(t, n, k') + W_n(c_{k'}, S_{n'}, t+|c_{k'}|)\}. \quad (9)$$

Now we are ready to present our strategy to solve the maximum-weight path problem in G. We proceed in stages from stage 0 to stage T. At each stage t we do not compute the maximum-weight paths ending at the nodes on layer t, but rather the maximum-weight paths ending at nodes on later layers, whose last visited node is on layer t. In other words, at stage t we compute the values $W(c_{k'}, S_{n'}, t+|c_{k'}|)$ for all k', n' such that $(c_{k'}, S_{n'}) \in \mathcal{C}$. These computations are organized in the following three steps:

1) Compute $W_n(c_{k'}, S_{n'}, t+|c_{k'}|)$ according to (8) for all $1 \le n' \le N$, $1 \le k' \le K$, such that $(c_{k'}, S_{n'}) \in \mathcal{C}$, and $n \in \alpha^{-1}(n', k')$. Note that the quantities $W(c_k, S_n, t)$ needed in (8) are known because they have been computed at previous stages.

2) Compute $\omega(t, n, k')$ defined in (6), for all $1 \leq n' \leq N$, $1 \leq k' \leq K$, and $n \in \alpha^{-1}(n', k')$.
3) Compute $W(c_{k'}, S_{n'}, t + |c_{k'}|)$ for all $1 \leq n' \leq N$, and $1 \leq k' \leq K$, using relation (9).

Next we evaluate the number of operations required at each step.

Step 1) In order to solve (8) for given k', n', n, we need $O(|\{k|(c_k, S_n) \in \mathcal{C}\}|)$ time. Summing over all n', k', n, and using the fact that n' is determined when n and k' are given, we obtain the total time complexity

$$O(\sum_{k'=1}^{K}\sum_{n=1}^{N}|\{k|(c_k, S_n) \in \mathcal{C}\}|) = O(\sum_{k'=1}^{K}|\mathcal{C}|) = O(K|\mathcal{C}|).$$

Step 2) At this step we need to compute $\omega(t, n, k')$ defined in (6), for all $1 \leq n \leq N$, $1 \leq k' \leq K$. Recall that the bit sequence $b_1 b_2 \cdots b_{\beta|c_{k'}|}$ appearing in (6) is the output of the sequence of transitions of the CC, starting in state S_n, with sequence of input bits $c_{k'}$. Since the channel is memoryless it follows that:

$$\omega(t, n, k') = \textstyle\sum_{j=1}^{\beta|c_{k'}|} \log_2 P_{ch}(y_{t\beta+j}|b_j) =$$

$$H(y_{t\beta+1} \cdots y_{t\beta+\beta|c_{k'}|}, b_1 b_2 \cdots b_{\beta|c_{k'}|}) \times$$

$$\log_2(p_e/(1-p_e)) + \beta|c_{k'}| \log_2(1-p_e),$$

where $H(\cdot, \cdot)$ denotes the Hamming distance between two bit sequences of the same length (i.e., the number of positions where the bits differ), and p_e denotes the channel bit error rate (assuming the channel is a binary symmetric channel). In order to find the Hamming distance in the above expression, $O(\beta|c_{k'}|)$ bit-level operations are needed. Therefore, to compute all the values $\omega(t, n, k')$, for $, 1 \leq k' \leq K$,$1 \leq n \leq N$, the number of operations required is

$$O(\textstyle\sum_{k'=1}^{K}\sum_{n=1}^{N}\beta|c_{k'}|) = O(\beta N \sum_{k'=1}^{K}|c_{k'}|).$$

This quantity can amount to $O(NK^2)$ if the source code tree is very unbalanced (for example for the Golomb-Rice code). We show however that all the values $\omega(t, n, k')$ can be computed more efficiently by using the source code tree structure.

Let L be some arbitrary node in the source code tree. Hence L can be an internal node, i.e., a prefix L_j of some source codeword, or L could be a leaf, i.e., some codeword c_k. Let $b'_1 b'_2 \cdots b'_{\beta|L|}$ be the sequence of bits output by the sequence of transitions in the state diagram of the CC starting from the state S_n with input bits L. Denote by $h(t, n, L) \triangleq H(y_{t\beta+1} \cdots y_{t\beta+\beta|L|}, b'_1 b'_2 \cdots b'_{\beta|L|})$.

By convention, $h(t,n,L)=0$ if L is the root of the source code tree (hence L corresponds to the empty bit sequence). If L' is the parent of the node L, then clearly $|L'|=|L|-1$ and $b'_1b'_2\cdots b'_{\beta(|L|-1)}$ is the sequence of bits output by the sequence of transitions in the state diagram of the CC starting from the state S_n with input bits L'. It follows that

$$h(t,n,L)=h(t,n,L')+$$

$$H(y_{t\beta+\beta(|L|-1)+1}\cdots y_{t\beta+\beta|L|},b'_{\beta(|L|-1)+1}\cdots b'_{\beta|L|}).$$

Consequently, $h(t,n,L)$ can be computed from $h(t,n,L')$ with β bit-level operations. Therefore, in order to obtain the values $h(t,n,c_{k'})$ for given n, we compute $h(t,n,L)$ for all nodes of the tree L, starting from the root L_1 and proceeding level by level until all nodes are exhausted. Thus, the total number of operations to find $h(t,n,c_{k'})$ for given n and all k' is proportional to the number of nodes of the tree, i.e., $O(K)$. Summing up for all $n, 1\le n\le N$, we obtain the time complexity of Step 2 to be $O(KN)$.

Step 3. The number of operations required to solve equation (9) is proportional to the size of the set $\alpha^{-1}(n',k')$. Therefore the total number of operations at Step 3 is in the order of

$$\textstyle\sum_{(n',k')\in\mathcal{C}}|\alpha^{-1}(n',k')|=$$

$$|\{(n,k',n')|(n',k')\in\mathcal{C},n\in\alpha^{-1}(n',k')\}|=$$

$$|\{(n,k',n')|(n',k')\in\mathcal{C},n'=\alpha(n,k')\}|\le$$

$$|\{(n,k')|1\le n\le N,1\le k'\le K\}|=KN.$$

Consequently, the time complexity of Step 3 becomes $O(KN)$.

From the above evaluation we see that the time complexity to process each stage is dominated by the complexity of Step 1. In the next section we prove that Step 1 can be sped up by a factor of K for Markov sources with the Monge property.

V. Efficient Decoding Using Fast Matrix Search

The complexity reduction of Step 1) presented in this section is based on the fast matrix search technique introduced in [1] for finding all column maxima in a totally monotone matrix.

For each n, we compute together the values $W_n(c_{k'},S_{\alpha(n,k')},t+|c_{k'}|)$ for all $k', 1\le k'\le K$. Let us fix some n. Assume first the case when the state-space size of the CFSM cannot be

reduced, hence, all pairs (c_k, S_n), $1 \leq k \leq K$, are in $\mathcal{C}$. Consider now the $K \times K$ matrix $A_{n,t}$ with elements $A_{n,t}(k,k')$ defined as follows

$$A_{n,t}(k,k') \triangleq W(c_k, S_n, t) + \log_2 P(a_{k'}|a_k) \tag{10}$$

for all $1 \leq k, k' \leq K$. Then relation (8) is equivalent to

$$W_n(c_{k'}, S_{\alpha(n,k')}, t + |c_{k'}|) = \max_{1 \leq k \leq K} A_{n,t}(k,k'), \tag{11}$$

in other words, $W_n(c_{k'}, S_{\alpha(n,k')}, t + |c_{k'}|)$ is the maximum element on the (k')-th column of the matrix $A_{n,t}$. Solving relation (11) for all pairs $(k', \alpha(n,k'))$ is equivalent to finding the maxima of all columns of the matrix $A_{n,t}$. The straightforward solution to this problem requires a number of operations proportional to the number of matrix elements, i.e. $O(K^2)$. However, if the matrix $A_{n,t}$ is so-called totally monotone, then the problem of all column maxima can be solved in $O(K)$ time by the fast matrix search algorithm, nicknamed SMAWK, introduced in [1].

The matrix $A_{n,t}$ is said to be totally monotone [1] with respect to the column maxima if the following relation holds:

$$A_{n,t}(k_1, k'_1) \leq A_{n,t}(k_2, k'_1) \Rightarrow$$
$$A_{n,t}(k_1, k'_2) \leq A_{n,t}(k_2, k'_2),$$
$$\text{for all } k_1 < k_2, k'_1 < k'_2.$$

A sufficient condition for the total monotonicity of the matrix $A_{n,t}$ is [1]

$$A_{n,t}(k_1, k'_2) + A_{n,t}(k_2, k'_1) \leq$$
$$A_{n,t}(k_1, k'_1) + A_{n,t}(k_2, k'_2),$$
$$\text{for all } k_1 < k_2, k'_1 < k'_2, \tag{12}$$

also known as the concave Monge condition [3]. By replacing the matrix elements from (10) in the above inequality we obtain that (12) is equivalent to

$$W(c_{k_1}, S_n, t) + \log_2 P(a_{k'_2}|a_{k_1}) + W(c_{k_2}, S_n, t) +$$
$$\log_2 P(a_{k'_1}|a_{k_2}) \leq W(c_{k_1}, S_n, t) + \log_2 P(a_{k'_1}|a_{k_1}) +$$
$$W(c_{k_2}, S_n, t) + \log_2 P(a_{k'_2}|a_{k_2}),$$
$$\text{for all } k_1 < k_2, k'_1 < k'_2,$$

which, after making the cancelations, becomes identical to relation (1). In conclusion, if the Markov source satisfies the Monge property (1), then all matrices A_n, $1 \leq n \leq N$, are totally monotone, hence the computations at Step 1 (for all n) complete in $O(KN)$ time.

Let us treat now the case when the number of pairs $(c_k, S_n) \in \mathcal{C}$ for given n, is less than K. Denote this number by m_n and assume that $m_n < K$. Then the matrix A_n has dimension $m_n \times K$, but it is still totally monotone if condition (1) is satisfied. As proved in [1], the matrix search problem can then be solved in $O(m_n(1 + \log(K/m_n)))$ time. Summing over all n we obtain

$$\sum_{n=1}^{N} m_n(1 + \log_2 \tfrac{K}{m_n}) =$$

$$\sum_{n=1}^{N} m_n(1 + \log_2 \tfrac{K}{|\mathcal{C}|} + \log_2 \tfrac{|\mathcal{C}|}{m_n}) =$$

$$(\sum_{n=1}^{N} m_n)(1 + \log_2 \tfrac{K}{|\mathcal{C}|}) + |\mathcal{C}| \sum_{n=1}^{N} \tfrac{m_n}{|\mathcal{C}|} \log_2 \tfrac{|\mathcal{C}|}{m_n}.$$

Since we have $\sum_{n=1}^{N} m_n = |\mathcal{C}|$, it follows that $\sum_{n=1}^{N} \frac{m_n}{|\mathcal{C}|} \log_2(|\mathcal{C}|/m_n) \leq \log_2 N$. Consequently, the time complexity of Step 1 is $O(|\mathcal{C}|(1+\log(KN/|\mathcal{C}|))$. Now Step 1 becomes the less extensive and the time complexity at each stage is dominated by Steps 2 and 3, hence it is $O(KN)$.

In conclusion, for Markov sources with the Monge property (1) the computations at each stage of the algorithm can be performed in $O(KN)$ time. Since there are $O(T)$ stages in total, it follows that the JSC MAP sequence decoding completes in $O(TKN)$ time.

VI. Accelerated Soft Output Max-Log-MAP Decoding

In some applications the decoder might need to provide the reliability information for each decoded symbol or bit. These soft outputs could either accompany the hard decisions or constitute the only output of the decoder, as is the case in constituent decoders in iterative JSC decoding schemes.

In this section we consider the case when bit-level soft outputs are desired. Note that the bit sequence $\mathbf{v} = v_1 \cdots v_T$ output by the SC is completely determined by the Markov sequence $\mathbf{u}$ and vice-versa. Optimum bit-level soft output decoders provide the a posteriori probabilities (APPs) $P(V_t = b|\mathbf{y}) = \sum_{\mathbf{v}, v_t = b} P(\mathbf{v}|\mathbf{y})$, for all $1 \leq t \leq T$ and $b \in \{0, 1\}$, where V_t denotes the random variable representing the t-th bit input to the CC. These APPs can be obtained by

running the BCJR algorithm [4] on the bit-level trellis obtained by expanding in time the state diagram of the CFSM which describes the chain MS$\rightarrow$SC$\rightarrow$CC.

Soft output decoders based on the approximation $\log_2(\gamma_1 + \cdots + \gamma_m) \approx \max_{1\leq i\leq m} \log_2 \gamma_i$, also known as Max-Log-MAP algorithms, are suboptimal, but more efficient alternatives for the calculation of the APPs [15]. The values to be computed are

$$\mathcal{Z}_t(b) \triangleq \max_{\mathbf{v}, v_t=b} \log_2 P(\mathbf{v}, \mathbf{y}), \tag{13}$$

for all $b \in \{0, 1\}$ and $1 \leq t \leq T$. Then the APPs are approximated as

$$P(V_t = b|\mathbf{y}) \approx \frac{\mathcal{Z}_t(b)}{\mathcal{Z}_t(0) + \mathcal{Z}_t(1)},$$

The quantities (13) can be evaluated in a Viterbi-like decoding manner in a forward and a backward pass through the bit-level trellis. Thus the asymptotical time complexity of the JSC decoding algorithm becomes $O(TK^2N)$. We show next that for Markov sources satisfying the Monge property, the running time can be reduced again by a factor of K by using the fast matrix search of [1].

Toward our goal note first that the quantity $\mathcal{Z}_t(b)$ equals the weight of the maximum-weight path in the graph G among all paths corresponding to bit sequences $\mathbf{v}$ whose t-th bit is b. In order to characterize such paths we need to introduce some more notations. For any t and any internal node L_j of the SC tree, other than the root, define $\mathcal{N}(t, L_j)$ as the set of all graph nodes $(c_k, S_n, t + |c_k| - |L_j|)$, for all n and all k such that L_j is a prefix of c_k. Moreover, let $\mathcal{N}(t, c_k)$ denote the set of all nodes (c_k, S_n, t). It is easy to see that a path P in G corresponds to a sequence $\mathbf{v}$ whose t-th bit is b, if and only if the path visits some node $n \in \mathcal{N}(t, L_jb)$ for some internal node L_j in the SC tree. Recall that L_jb denotes the concatenation between the bitsequence describing L_j and the bit b. Thus L_jb is either an internal node different from the root, or a codeword. Moreover, the maximum-weight path among all paths visiting some node ν is the concatenation of the maximum-weight path from the source to ν and the maximum-weight path from ν to a final node. For any node ν, let $\bar{W}(\nu)$ denote the weight of the maximum-weight path among all paths in the graph from node ν to a final node. Then we have

$$\mathcal{Z}_t(b) \;\; = \;\; \max_{1\leq j\leq K-1} \max_{\nu\in\mathcal{N}(t,L_jb)} (W(\nu) + \bar{W}(\nu)). \tag{14}$$

For any t and any L corresponding to a codeword or an internal node of the SC tree other than the root, define $\gamma(t, L)$ as

$$\gamma(t, L) \triangleq \max_{\nu \in \mathcal{N}(t,L)} (W(\nu) + \bar{W}(\nu)).$$

Hence, (14) becomes

$$\mathcal{Z}_t(b) \;\; = \;\; \max_{1 \le j \le K-1} \gamma(t, L_j b). \tag{15}$$

Notice that for $2 \le j \le K-1$, we have $\mathcal{N}(t, L_j) = \mathcal{N}(t+1, L_j 0) \cup \mathcal{N}(t+1, L_j 1)$, which implies that

$$\gamma(t, L_j) = \max\{\gamma(t+1, L_j 0), \gamma(t+1, L_j 1)\}. \tag{16}$$

On the other hand,

$$\gamma(t, c_k) = \max_{n, (c_k, S_n) \in \mathcal{C}} (W(c_k, S_n, t) + \bar{W}(c_k, S_n, t)). \tag{17}$$

We are ready now to present the efficient Max-Log-MAP JSC decoding for Markov sequences with the Monge property. The algorithm consists of two passes, a forward pass and a backward pass. The forward pass proceeds as in the MAP sequence decoding, computing and storing the values $W(\nu)$ for all graph nodes ν. Using the strategy described in Sections 5 and 6, this pass completes in $O(TKN)$ operations.

During the backward pass, the values $\bar{W}(\nu)$ and $\mathcal{Z}_t(b)$ are calculated, processing the graph layers $t, 1 \le t \le T$, in decreasing order. Precisely, the computations associated to each layer t, are divided into the following steps

1) Find $\bar{W}(\nu)$ for all nodes ν at layer t.
2) Compute the quantities $\gamma(t, L)$ for all $L = L_j$, $2 \le j \le K-1$ using (16), and for all $L = c_k$, $1 \le k \le K$, using (17).
3) Evaluate $\mathcal{Z}_t(b)$ according to (15).

Step 2 completes in $O(\mathcal{C})$ time since solving (16) for all L_j needs $O(K)$ operations, while solving (17) for all c_k necessitates $O(\mathcal{C})$ operations. The running time of step 3 is clearly $O(K)$.

Step 1 can be performed efficiently in $O(KN)$ time by exploiting the Monge property, using a similar idea to the one developed in Section 6. Precisely, we form the $K \times m_n$ matrix $B_{n,t}$,

with elements $B_{n,t}(k',k)$ defined as follows

$$B_{n,t}(k',k) \triangleq \bar{W}(c_{k'}, S_{\alpha(n,k')}, t+|c_{k'}|) + \log_2 P(a_{k'}|a_k) + \omega(t,n,k'),$$

for all $1 \leq k' \leq K$ and c_k such that $(c_k, S_n) \in \mathcal{C}$. Then, finding all $\bar{W}(c_k, S_n, t)$ for given t and n, reduces to the problem of finding all column maxima of the matrix $B_{n,t}$. Moreover, it can be easily verified that matrix $B_{n,t}$ satisfies the Monge property (12) when (1) holds. Therefore the SMAWK algorithm of [1] can be applied to find all its column maxima in $O(K)$ time. Thus, the total time to perform Step 1 is $O(KN)$ as we claimed. This leads to an $O(TKN)$ running time of the backward pass, and hence of the Max-Log-MAP JSC decoding algorithm as well.

Before ending this section we point out that the complexity reduction result extends trivially to the case when the CC is not present, by considering $N = 1$. In this case the efficient bit-level soft output Max-Log-MAP JSC decoding algorithm for Markov source satisfying the Monge property, completes in $O(TK)$ time, compared to $O(TK^2)$ decoding time when the Monge property is not exploited.

The strategy highlighted in this work easily extends to the case when the source codewords have variable length and the Markov sequence size I is known to the decoder. The decoder graph is more complex since each node is attached a symbol counter too, leading to an $O(TIK^2N)$ time MAP sequence JSC decoding algorithm when the Monge property is not exploited. Knowledge of the number of input symbols to the decoder facilitates the symbol-by-symbol soft output JSC decoding, too. Both bit-level and symbol-level Max-Log-MAP JSC decoding require $O(TIK^2N)$ operations as well. However, for Markov sequences satisfying the Monge property, the complexity reduction technique used in the previous sections can be adapted to the new decoder graph, leading to a decrease by a factor of K in the time complexity of the JSC decoding.

VII. Conclusion

This paper addresses the problem of joint source-channel (JSC) decoding of a Markov sequence which is encoded by a source code followed by a convolutional code, and sent through a noisy memoryless channel. It is proved that for Markov sources satisfying the so-called Monge property, both the maximum a posteriori probability (MAP) sequence decoding, as well as the soft output

Max-Log-MAP decoding can be expedited by a factor of K without penalizing the performance, where K is the size of the Markov source alphabet. The strategy used to increase the speed is a convenient organization of computations at the decoder combined with a fast matrix search technique enabled by the Monge property. Moreover, the same complexity reduction holds in the case of soft output Max-Log-MAP JSC decoding in the case when the convolutional coder is absent, result which was not known previously.

REFERENCES

[1] A. Aggarwal, M. Klawe, S. Moran, P. Shor, and R. Wilber, "Geometric applications of a matrix-searching algorithm", *Algorithmica*, 2(1987), pp.195-208.

[2] F. Alajaji, N. Phamdo, N. Farvardin and T. Fuja, "Detection of binary Markov sources over channels with additive Markov noise", *IEEE Trans. Inform. Th.*, vol. 42, no. 1, pp. 230-239, Jan. 1996.

[3] A. Apostolico and Z. Galil(eds.), *Pattern Matching Algorithms*, New York 1997.

[4] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", *IEEE Trans. Inform. Theory*, vol. 20, no. 2, pp. 284-287, Mar. 1974.

[5] R. E. Burkard, B. Klinz, and R. Rudolf, Perspectives of Monge properties in optimization, *Discrete Applied Mathematics* 70, 1996, pp. 95-161.

[6] Q. Chen and K. P. Subbalakshmi, "An Integrated Joint Source-Channel Decoder for MPEG-4 Coded Video", *Proc. IEEE Vehicular Tech. Conf. 2003*, pp. .

[7] N. Demir, K. Sayood, "Joint source/channel coding for variable length codes", *Proc. DCC'98 Data Compression Conference*, pp. 139-148, 1998.

[8] S. Dumitrescu and X. Wu, "On the complexity of joint source-channel decoding of Markov sequences over memoryless channels", to appear in *IEEE Trans. Communications*. An extended abstract appeared in *Proc. ISIT'05*.

[9] J. Garcia-Frias and J. D. Villasenor, "Joint turbo decoding and estimation of Hidden Markov sources", *IEEE J. on Selected Areas in Communications*, vol. 19, no. 9, pp. 1671-1679, Sept. 2001.

[10] A. Guyader, E. Fabre, C. Guillemot, and M. Robert, "Joint source-channel turbo decoding of entropy-coded sources", *IEEE Journal on Selected Areas in Comm.*, vol. 19, no. 9, pp. 1680-1696, Sept. 2001.

[11] J. Hagenauer and N. Gortz, "The Turbo Principle in Joint Source-Channel Coding", *IEEE ITW 2003*, pp. 275-278, April 2003.

[12] A. H. Murad and T. E. Fuja, "Joint source-channel decoding of variable-length encoded sources", *IEEE ITW'98*, June 1998, pp. 94-95.

[13] M. Park and D. J. Miller, "Joint source-channel decoding for variable-length encoded data by exact and approximated MAP sequence estimation", *IEEE Trans. Comm.*, vol. 48, no. 1, pp. 1-6, Jan. 2000.

[14] N. Phamdo and N. Farvardin, "Optimal Detection of Discrete Markov Sources over Discrete Memoryless Channels - Applications to Combined Source-Channel Coding", *IEEE Trans. Inf. Th.*, vol. 40, no.1, pp. 186-193, Jan. 1994.

[15] P. Robertson, E. Villebrun and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain", *Proc. IEEE Int. conf. Commun.*, Seattle, WA, pp.1009-1013, Jun. 1995.

[16] K. Sayood and J. C. Borkenhagen, "Use of Residual Redundancy in the Design of Joint Source/Channel Coders", *IEEE Trans. Comm.*, vol. 39, no. 6, pp. 835-846, June 1991.

[17] K. P. Subbalakshmi and J. Vaisey, "On the Joint Source-Channel Decoding of Variable-Length Encoded Sources: The BSC Case", *IEEE Trans. Comm.*, vol. 49, no. 12, pp. 2052-2055, Dec. 2001.

[18] X. Wu, S. Dumitrescu, and Z. Wang, "Monotonicity-based fast algorithm for MAP estimation of Markov sequences over noisy channels", *IEEE Trans. Inf. Theory*, vol. 50, pp. 1539-1544, July 2004.