OPTIMAL PACKETIZATION OF VLC AND CONVOLUTION CODED MARKOV SEQUENCES

Xiaohan Wang and Xiaolin Wu

Department of Electrical and Computer Engineering, McMaster University
Hamilton, Ontario, Canada, L8S 4K1
xhwang@grads.ece.mcmaster.ca / xwu@mail.ece.mcmaster.ca

ABSTRACT

We consider the problem of packetizing a variable length coded Markov sequence into fixed length packets, while being protected by variable rate channel code. Given the total transmission bit budget, a joint source-channel coding problem is how to partition the input sequence and how to determine the coding rates of individual packets for minimum expected distortion when the sequence is sent over binary symmetric channel. Three methods are proposed to estimate the performance of a sequence when transmitted through the system, based on which we convert the joint source-channel coding problem into a shortest path problem in a weighted directed acyclic graph which can be solved by using dynamic programming. Simulation shows that the overall performance of the system can be improved by 10-30% compared with the performance of the fixed rate packetization scheme.

1. INTRODUCTION

Maximum *a posteriori* (MAP) decoding is a joint source-channel decoding technique that exploits the residual redundancy of a source code stream. A common structure of source redundancy to be exploited by a MAP decoder is that of a Markov sequence. If a Markov sequence is coded by a fixed-length code the MAP decoding is quite straightforward. The problem gets more complex if the source code is of variable length. This is because channel errors can easily cause loss of synchronization on a variable-length code (VLC). Since most entropy codes are of variable length, MAP decoding of VLC Markov sequence is of greater practical interest and importance.

The idea of MAP decoding seems to contradict Shannon's classic separation theorem [1], in which known memory of the source should be utilized thoroughly by entropy coding, and the input sequence of the channel encoder is always assumed memoryless. However, situations exist where leaving the memory in the source is more practical and advantageous to the overall performance of the system [2].

Data packetization is a must in packet switched networks. Besides making efficient and flexible use of the network, it provides a means of error resilience. The physical boundaries of N packets can prevent the propagation of loss of synchronization. MAP makes use of the correlations between symbols to correct errors. Setting packet boundaries at locations where the Markov transition probability is small and hence the error correcting capability is weak, can improve the performance of MAP decoding. In a coding perspective, the packet header, which consumes bits to encode the packet number (sequencing information), the number of source symbols and the channel code rate for the packet, is redundancy in the form of side information. This redundancy should be exploited by the joint source-channel decoding process. Therefore, MAP and packetization can work jointly to achieve better overall performance.

To be more practical, this paper considers the scenario where all packets are protected by a variable rate channel code after the packetization. All packets are then sent through a binary symmetric channel (BSC). Given a source code, a set of channel codes with different rates, a BSC, the total number of packets and the length of a single packet, we aim to find the optimal packtization scheme that minimizes the decoding errors.

This paper is organized as follows. Section 2 gives the formal formulation of the problem. Two graph based representations and solutions of the problem are given in Section 3. Section 4 discusses some methods of the estimation of error propagation length. The experimental results are presented and analyzed in Section 5.

2. PROBLEM FORMULATION

We consider the following packetized communication scenario. An input sequence of M symbols $\mathbf{x} = x_0 x_1 \cdots x_{M-1}$, is generated from a first-order discrete Markov source with alphabet $\mathcal{A} = \{\alpha_0, \alpha_1, \cdots, \alpha_{K-1}\}$. The Markov source is characterized by transition probabilities $p(\alpha_j|\alpha_k) = Pr(x_t = \alpha_j | x_{t-1} = \alpha_k)$ for all $\alpha_j, \alpha_k \in \mathcal{A}$, and the prior probability distribution $p(\alpha_k) = Pr(x_0 = \alpha_k)$, $\alpha_k \in \mathcal{A}$. Before transmission, the input sequence $\mathbf{x}$ is partitioned into N subsequences:

$$\mathbf{x} = x_0 \cdot\cdot x_{m_1-1}, x_{m_1} \cdot\cdot x_{m_2-1}, \cdots, x_{m_{N-1}} \cdot\cdot x_{M-1}$$

0-7803-9332-5/05/$20.00 ©2005 IEEE

one for each of the N packets. For the sake of cleaner notation, we write $x[m_i, m_{i+1}) = x_{m_i} \cdot x_{m_{i+1}-1}$, where $m_0 = 0$ and $m_N = M$. The vector $\mathbf{m} = (m_1, m_2, \cdots, m_{N-1})$ is called packetization vector because it uniquely determines a packetization scheme of $\mathbf{x}$. Note that in the above formulation we require each packet to contain an integer number of source symbols.

Given a packetization $\mathbf{m}$, each subsequence $x[m_i, m_{i+1})$ is encoded by a VLC source encoder, such as a Huffman encoder, whose codebook is $\mathcal{C} = \{c_0, c_1, \cdots, c_{K-1}\}$ with c_k for α_k. K is the number of codewords and the length (in bits) of a codeword c_k is denoted by $|c_k|$. When the input is $x[m_i, m_{i+1})$, the output of the source encoder is $b[m_i, m_{i+1})$. For convenience, we will use $\mathbf{x}_i$ and $\mathbf{b}_i$ as shorthands for $x[m_i, m_{i+1})$ and $b[m_i, m_{i+1})$ in the sequel of this paper. The length (in bits) of $\mathbf{b}_i$ is expressed by $|\mathbf{b}_i|$.

All the VLC-coded sequences $\mathbf{b}_i$, $i = 0, \cdots, N-1$, are then protected by a variable rate channel code (such as a variable rate convolutional code), with rate $r[m_i, m_{i+1}) \in \mathcal{R}$, where $\mathcal{R} = \{R_1, R_2, \cdots, R_B\}$ is the set of all available channel code rates. We use r_i as a shorthand for $r[m_i, m_{i+1})$, and call $\mathbf{r} = (r_1, r_2, \cdots, r_N)$ the channel rate vector. After $\mathbf{b}_i$ is channel encoded, the length of the output bit stream is $|\mathbf{b}_i|/r_i$.

These N source-channel encoded bit streams are placed into N packets. If the length of each packet is L bits, we have the constraint $|\mathbf{b}_i|/r_i \le L$ for all i's. All N packets are then transmitted through a BSC, whose crossover probability is p_c. At the receiver, we do channel decoding followed by MAP decoding on each packet and obtain N decoded subsequences $\mathbf{y}_i$. Finally the receiver concatenates all $\mathbf{y}_i$'s into an output sequence $\mathbf{y}$. We measure the performance of such a system by the expected total error propagation length (EPL) between the system input $\mathbf{x}$ and output $\mathbf{y}$, denoted by $\bar{D}(\mathbf{x}, \mathbf{y})$, where the total EPL measures the number of symbols that are decoded incorrectly after $\mathbf{x}$ and $\mathbf{y}$ are aligned by minimum edit distance.

Ideally, one wants to minimize $\bar{D}(\mathbf{x}, \mathbf{y})$ over all possible packetization vectors $\mathbf{m}$ and the channel rate vectors $\mathbf{r}$, given the VLC, $\mathbf{x}$, and p_c. Since our system imposes the condition that an integer number of source symbols are packed into a packet, the influence of any bit error is confined to the boundary of the packet. This makes $\bar{D}(\mathbf{x}, \mathbf{y})$ an additive measure, namely,

$$\bar{D}(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{N-1} E(D(\mathbf{x}_i, \mathbf{y}_i)). \tag{1}$$

Since the distortion $D(\mathbf{x}_i, \mathbf{y}_i)$ of the i^{th} subsequence is monotonically nonincreasing in channel coding rate r_i, given m_i, the optimal rate r_i is determined by

$$r_i^* = \min\{r \in \mathcal{R} \mid |\mathbf{b}_i|/r \le L\}. \tag{2}$$

Therefore, the packetization vector $\mathbf{m}$ becomes the only variable in the underlying optimization problem, given the VLC, $\mathbf{x}$, and p_c, and we can equivalently write

$$\bar{D}(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{N-1} \bar{D}(x[m_i, m_{i+1})), \tag{3}$$

where $\bar{D}(x[m_i, m_{i+1}))$ is the expected sum of error propagation lengths by source-channel coding of the subsequence $x[m_i, m_{i+1})$ in one packet, where the channel code rate is selected as r_i^*.

Finally, we can state the problem of optimal packetization of source-channel coded Markov sequence $\mathbf{x}$, given the variable length source code C, BSC crossover probability p_c, packet size L, and the number of packets N, as the following

$$\min_{\mathbf{m}} \sum_{i=0}^{N-1} \bar{D}(x[m_i, m_{i+1})) \tag{4}$$

3. ALGORITHM DEVELOPMENT

To facilitate the development of algorithms for the problem given in (4), a weighted directed acyclic graph (DAG) G is constructed as follows. There are $M+1$ vertices in G, denoted v_i, in which v_i corresponds to the symbol x_i for i from zero to $M-1$ and v_M to the end of sequence $\mathbf{x}$. For any subsequence $x[i, j)$, $i < j$, if $|b[i, j)|$ is less than L, there exists a directed edge (i, j) that leaves vertex v_i and enters vertex v_j. The weight of edge (i, j) is $\bar{D}(x[i, j))$. Then the optimal solution can be found by searching for the shortest path from v_0 to v_M that has exact N edges [3,4].

Specifically, for each vertex v_i, a list of weights $\omega(i, n)$, $n = 0, 1, \cdots, N$, are saved, which give the solution of the subproblem where the input sequence is $x[0, i)$ and the number of packets is n. Then $\omega(j, n)$ can be calculated by

$$\omega(j, n) = \min_{\substack{0 \le i < j, \\ \text{if edge } (i,j) \text{exists}}} \{\bar{D}(x[i, j)) + \omega(i, n-1)\}. \tag{5}$$

The value of $\omega(j, n)$ and the survival path are recorded. To initialize, we have $\omega(0, 0) = 0$ and $\omega(i, 0) = +\infty$ for $i = 1, 2, \cdots, M$.

After all $\omega(i, n)$'s are calculated, we start from $\omega(M, N)$ and trace back along the survival paths. The obtained path gives the optimal solution. Because there are M vertices in G, the out-degree of each vertex is $O(L)$, and for each vertex, N values need to be calculated, the complexity of the above dynamic programming algorithm is $O(MNL)$.

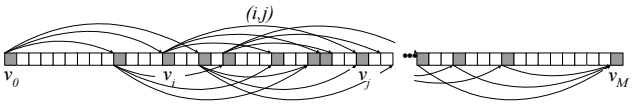

Fig. 1. Graph representation of the greedy search algorithm.

It is possible to speed up the algorithm by removing edges of large weights from graph G. Instead of considering all possible outgoing edges from a given vertex v_i, for each available rate $r \in \mathcal{R}$, we only include the edge

that crosses the longest sequence subject to the constraint of packet size. Only those vertices that have incoming edges (called activated vertices, as gray shaded in Fig. 1) can emit outgoing edges. This graph construction algorithm is given below.

Algorithm *Graph Construction for Greedy Search*

1. activate v_0
2. **for** $i = 0, 1, \cdots, M-1$
3. **do if** v_i is activated
4. **then for** $t = 1, 2, \cdots, B$
5. **do** $j \leftarrow \max\{k | k \le M, b[i,k)/R_t \le L\}$
6. add a new edge (i, j)
7. **if** v_j is non-activated
8. **then** activate v_j

The resulting graph will be much sparser than the complete DAG. The same dynamic programming method can be used on the reduced graph. Let the average out-degree of vertices be B, the complexity of the search algorithm is reduced to $O(MNB)$.

4. ESTIMATION OF ERROR PROPAGATION LENGTH

4.1. Error Analysis

The remaining challenge to solving the optimal packetization problem (4) is how to compute $\bar{D}(x[m_i, m_{i+1}))$. It is well known that with proper interleaver/deinterleaver and channel encoder/decoder a BSC can be regarded as a new BSC of reduced crossover probability. The new crossover probability p_e is determined by p_c, r_i and the structure of the channel code and can be obtained by simulations. Let $l_i(j)$ be the error propagation length caused by flipping only the j^{th} bit of $\mathbf{b}_i$. Empirically we observed that the random variable $l_i(j)$ obeys a geometric distribution whose parameter λ, is determined by p_e and the MAP decoding process.

One can show that the average interval between two consecutive bit errors for BSC is $\frac{1}{p_e}$. The expectation of $l_i(j)$ is $\frac{1}{\lambda} - 1$. In practice, $\frac{1}{p_e} \gg \frac{1}{\lambda} - 1$ holds. Therefore, if more than one bits flip during the transmission, the probability that the bit errors interfere with each other will be very small. Thus, the expected total EPL of sequence $x[m_i, m_{i+1})$ can be approximated by

$$\bar{D}(x[m_i, m_{i+1})) \approx p_e \sum_{j=0}^{|\mathbf{b}_i|-1} l_i(j). \tag{6}$$

The next step is to compute $l_i(j)$ for all j. Three different methods of calculating $l_i(j)$ are discussed in the following subsections.

4.2. Direct Method

From definition, we can calculate $l_i(j)$'s directly by using the following algorithm.

Algorithm *Direct Method*

1. Calculate p_e from p_c, r_i^* and channel code structure
2. **for** $j = 0, 1, \cdots, |\mathbf{b}_i| - 1$
3. **do** flip the j^{th} bit of $\mathbf{b}_i$, get a new sequence $\mathbf{b}_i'$
4. MAP decode $\mathbf{b}_i'$ (by using p_e) into $\mathbf{y}_i$
5. $l_i(j) \leftarrow D(\mathbf{x}_i, \mathbf{y}_i)$

The complexity of MAP decoding in calculating $l_i(j)$ is $O(N^2|\mathbf{b}_i|)$, or $O(N|\mathbf{b}_i|^2)$ when the input sequence is Gaussian Markov [5]. Therefore, the complexity to calculate $\bar{D}(x[m_i, m_{i+1}))$ by direct algorithm is $O(N^2|\mathbf{b}_i|^2)$, or $O(N|\mathbf{b}_i|^2)$ when the input sequence is Gaussian Markov, which is very expensive when $|\mathbf{b}_i|$ is large. We propose two approximate methods to reduce the complexity.

4.3. Moving Window Method

In MAP decoding, the EPL of a single bit error is only dependent on those symbols that are not far away from the flipped bit. Thus we analyze sequence $\mathbf{x}_i$ in a window of $2W+1$ symbols and centered at the symbol of the flipped bit. Instead of MAP decoding of the whole sequence $\mathbf{b}_i$, we confine MAP to the subsequence of the window. As j changes from 0 to $|\mathbf{b}_i| - 1$, the window moves along the sequence $\mathbf{x}_i$ from left to right. Because W is a constant, the complexity to calculate $\bar{D}(x[m_i, m_{i+1}))$ decreases to $O(N^2|\mathbf{b}_i|)$, or $O(N|\mathbf{b}_i|)$ if the input sequence is Gaussian Markov [5].

4.4. Dictionary Method

In moving window method, if the width of the window is small enough, we can precompute the values for all combinations of the symbols in the window in advance and save the results in a dictionary. Thus, $l_i(j)$ can be had by table look up, reducing the complexity dramatically.

Now we show how the dictionary is created. Consider a window of $2w+1$ symbols centered at the current symbol. Denote the sequence in this window by $\mathbf{z} = z_{-w} \cdots z_0 \cdots z_w$. The value of $\mathbf{z}$ is used as a key to the dictionary. The entry for $\mathbf{z}$ is the expected value $\bar{l}(z_0)$ of the summation of the EPL's caused by the single bit errors at the bits of symbol z_0, which can be precomputed using a long training sequence $\mathbf{t} = t_0 t_1 \cdots t_T$ generated from the first-order discrete Markov source, by

$$\bar{l}(z_0) = E\Bigg[\sum_{\substack{0 \le i < T \\ t_{i-w} \cdots t_{i+w} = \mathbf{z}}} \sum_{j \in t_i} l_{\mathbf{t}}(j) \Bigg]. \tag{7}$$

Note that $l_{\mathbf{t}}(j)$ is calculated by using the moving window method within a bigger window of $2W+1$ symbols. With this dictionary, $\bar{D}(x[m_i, m_{i+1}))$ can be easily calculated.

This algorithm is very fast, with a complexity of only $O(|\mathbf{b}_i|)$. However, since p_e varies for different rates r_i, $l_{\mathbf{t}}(j)$ is a function of r_i. We need to create different dictionaries, one for each $r_i \in \mathcal{R}$. Since $|\mathcal{R}| = B$, the total table size is BK^{2w+1}. For sufficient accuracy, w cannot be too small, incurring huge memory use. Fortunately, the ta-

ble is very sparse due to the Markov property of $\mathbf{z}$. Hashing technique can be used to reduce the space requirement.

5. EXPERIMENTAL RESULTS

In our experiments the source is a scalar-quantized (uniform with nine code cells) zero-mean, unit-variance, first-order Gaussian-Markov process of correlation coefficient 0.9. The input sequences are Huffman encoded with nine codewords ($K = 9$). We use four different packet lengths ($L = 60, 120, 240, 384$ bits), where the last one is the length of an ATM packet, and three different number of packets ($N = 10, 20, 40$). The crossover probability of the BSC is p_c=0.005. We use convolutional code as the channel code. The set of all available rates is $\mathcal{R} = \{\frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, 1\}$. The constraint lengths for these rates are $3, 4, 3, 4$ and 0 respectively. We pick the width of the moving window to be 21 symbols ($W = 10$).

The length of the $\mathbf{z}$ in the dictionary method is five, i.e. $w = 2$. The total number of indices BK^{2w+1} equals to $5 \cdot 9^5 = 295245$, only 42906 of which are non-zero and are saved in the Hash table.

Table 1 gives the accuracy of the three methods in estimating the EPL, measured by the average relative error. The direct method and moving window method have almost the same accuracy, and are better than the dictionary method. It can also be seen that the accuracy is better for lower crossover probability, which is expected because the approximation in (6) is more accurate for lower crossover probability.

Table 1. Estimation errors of different methods.

M	Method	Crossover Probability p_c	
		0.003	0.001
20	Direct	5.25%	1.96%
	Moving Window	5.25%	1.96%
	Dictionary	15.36%	17.72%
60	Direct	4.90%	2.16%
	Moving Window	5.01%	2.27%
	Dictionary	13.27%	15.18%
120	Direct	5.62%	2.14%
	Moving Window	5.47%	2.80%
	Dictionary	10.67%	11.02%

We implemented the two search algorithms proposed in Section 3, and compared them with the fixed rate packetization, whose rate is $\frac{2}{3}$. For each (L, N) combination, we tested 100 input sequences. For fair comparison, we selected only the sequences such that with the fixed channel code rate the source symbols can be packed exactly without wasting packet payload. For each combination (L, N) and each input sequence, we compared fixed rate packetization and three versions of variable rate packetization: 1) moving window method with complete search, 2) moving window method with greedy search and 3) dictionary method with greedy search. For each method and each input sequence, we ran 5000 simulations. The simulation results are reported in Tables 2 and 3, whose entries are the reduction (in percentage) of $D(\mathbf{x}, \mathbf{y})$ over that of the fixed rate packetization.

Table 2. Results for different N ($L = 120$, $p_c = 0.005$).

N	Complete Search & Moving Window	Greedy Search & Moving Window	Greedy Search & Dictionary
10	24.08%	24.36%	23.30%
20	27.06%	27.03%	26.93%
40	29.71%	29.36%	29.10%

Table 3. Results for different L ($N = 10$, $p_c = 0.005$).

L	Complete Search & Moving Window	Greedy Search & Moving Window	Greedy Search & Dictionary
60	32.05%	31.54%	28.34%
120	24.08%	24.36%	23.30%
240	16.96%	16.21%	16.58%
384	12.62%	12.07%	10.84%

Referring to the tables, the result of greedy search is almost the same as that of complete search. The performance of the dictionary method with greedy search is only a little worse than that of moving window method with complete search, but the former is much faster than the latter.

When the number of packets (N) increases, so does the improvement. This is because the more packets, the more flexibilities exist in the optimization. In a coding perspective, more packets mean higher density of header overhead, hence greater level of redundancy to be exploited by the optimal packetization process. Conversely, as the size of the packets (L) increases, the less improvement is achieved. The reason is that the larger the packet size, the lower density of header information, hence less redundancy available to combat the channel errors.

6. REFERENCES

[1] C. E. Shannon, Collected Papers, edited by N.J.A. Sloane and A. Wyner, *IEEE press*, New York, 1993, p. 40.

[2] G. Buch, F. Burkert, J. Hagenauer and B. Kukla, "To compress or not to compress", *Global Telecommunications Conference, 1996. GLOBECOM '96*, p. 18-22, Nov. 1996.

[3] M. Park and D. J. Miller, "A joint source-channel decoding for variable-length encoded data by exact and approximate MAP sequence estimation", *IEEE Trans. Commun.*, vol. 48, Jan. 2000.

[4] Z. Wang, X. Wu, and S. Dumitrescu, "Length-Constrained MAP Decoding Revisited", *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, 2004

[5] X. Wu, S. Dumitrescu and Z. Wang, "Monotonicity-based Fast Algorithms for MAP Estimation of Markov Sequences over Noisy Channels", *IEEE Trans. Inform. Theory*, vol. 50, pp. 1539-1544, Jul. 2004.