

ASYMPTOTICALLY OPTIMAL VARIABLE-TO-FIXED LENGTH CODES FOR MARKOV SOURCES*

K. VISWESWARIAH, S. KULKARNI and S. VERDU

*Department of Electrical Engineering
Princeton University, Princeton, NJ 08544
Submitted: April 15, 1988*

We show universal variable-to-fixed length codes which achieve an optimal rate of convergence to the entropy. The redundancy of the code is $H \log \log M / 2 \log M$ per parameter where H is the entropy of the source and M is the number of codewords. This generalizes results known for binary i.i.d. sources to the class of Markov sources. The coding method requires storage of the dictionary which could be prohibitive to achieve low redundancies. We analyze the coding rate of a scheme that does not require the storage of dictionaries.

1. INTRODUCTION

We assume that the source X takes values in a finite set A . We code the source using a finite dictionary $\mathcal{D} \in A^m$ which is used to parse an infinite source sequence into variable length phrases. Each phrase is then coded using a fixed number of bits. The number of bits required to code a phrase is $\lceil \log |\mathcal{D}| \rceil$. We require that the dictionary be complete. A complete dictionary is one such that any infinite sequence will have a prefix in the dictionary. We also assume that the dictionary is prefix free. The dictionary being prefix free ensures that there is a unique way to parse every source string. For the dictionary \mathcal{D} to be a good source code it is required that the expected input phrase length be as high as possible for a fixed output size (i.e., a fixed dictionary size).

Tunstall (1967) gave an algorithm to find the dictionary of a given size that maximizes the expected phrase length for a given memoryless source. We require an efficient algorithm because the number of dictionaries with M leaves is

*This work was partially supported by the National Science Foundation under Grants NYI Award IRI-9457645 and NCR 9523805

exponential in M . The method requires the source distribution to be known. The problem of finding good dictionaries for coding Markov sources was investigated by Tjalkens and Willems (1987), and Savari and Gallager (1997), where codes which achieve the entropy rate for sources with memory were given. Tjalkens and Willems (1992) solved the universal coding problem for the class of binary valued and i.i.d. sources. They show that their coding method converges to the entropy at an optimal rate. Here a more general universal coding problem is investigated, the class of sources consists of Markov sources which take values in a finite set of size possibly greater than two.

In Section 2, the problem considered in this paper is formally defined. In Section 3, a variable-to-fixed source code is found which is shown to achieve a redundancy of $Hk(a-1) \log \log M/2 \log M$, where H is the entropy rate of the source, M is the size of the dictionary, a is the alphabet size and k is the number of states in the Markov source. In Section 4, a converse for the above achievability result is given by showing that the set of parameters for which a code can achieve a better redundancy than the code in Section 3 has vanishingly small volume. These results are a generalization of the results of Tjalkens and Willems (1992) where the optimal redundancy was given as $H \log \log M/2 \log M$ for binary, memoryless sources (i.e. $k=1$ and $a=2$). In Section 5, we investigate the performance of a implementable variable-to-fixed length coding method, which was proposed as an adaptive method for coding memoryless sources by Nissenbaum and Feder (1994).

2. PROBLEM FORMULATION

Let the source output alphabet be $A = \{1, 2, \dots, a\}$ and the set of states of the Markov source be $S = \{1, 2, \dots, k\}$. The Markov source is fully defined by the next-state function $N: A \times S \rightarrow S$ and the letter probability matrix $P(\cdot | \cdot)$ which is a matrix of size $A \times S$ and has entries belonging to $(0, 1)$. At each time the source outputs a letter from A and moves to another state. The next state depends on the present state and the output and is given by the next state function N . Let X_i, S_i be the random variables denoting the output and state respectively at time i . The distribution of the Markov source is defined by

$$Pr(X_n = x | S_n = s, X_{n-1}, S_{n-1}, \dots) = P(x | s).$$

The notation $P(x^* | s)$ for the probability of a string x^* occurring starting in the state s will be used. Note that this probability depends on P and N . Also note that Markov chains of a finite order are a special case of the Markov source as we

have defined it. Throughout it shall be assumed that the next state function is known to both the encoder and the decoder while $P(\cdot | \cdot)$ is unknown.

We will allow the encoder and the decoder to choose the dictionary to be used depending on what state the source is in. Both the encoder and the decoder can keep track of the state the source since we assume that the next state function is known. Thus an encoder \mathcal{D} is a collection of $|S|$ dictionaries. A dictionary \mathcal{D}_s is used to parse the source output if the source is in state s . Consider a collection of dictionaries \mathcal{D}_s with a maximum of M segments in a dictionary. The performance of this collection of dictionaries will be determined by the expected output length relative to $\log M$. To define the compression ratio precisely we first define a segment source as in Tjalkens and Willems (1987). The starting state of a segment and the segment determine the state at the start of the next segment. Also the probability of a segment occurring given the present state and the past states and segments depends only on the present state. Thus the segment source is a Markov source induced by the original Markov source and \mathcal{D} . Let X_1^*, X_2^*, \dots denote the output of the induced segment source. The state set S of the segment source need not be irreducible even if it is irreducible for the original source. Let $\mathbf{G} = \{G_r : r = 1, 2, \dots, R\}$ be the set of all the irreducible subsets of S . The performance of the code is governed by the expected average phrase length given by

$$E(L) = \lim_{n \rightarrow \infty} E \frac{1}{n} \sum_{i=1}^n l(x_i^*).$$

This is also given by (see Tjalkens and Willems, 1987)

$$E(L) = \sum_{i=1}^R Pr(G_i) \sum_{s \in G_i} q(s | G_i) \sum_{x^* \in \mathcal{D}_s} l(x^*) P(x^* | s)$$

where $Pr(G_i)$ is the probability of the source entering the irreducible set G_i , and $q(s | G_i)$ is the steady state probability of $s \in G_i$ given that the source has entered G_i . Our goal is to find a collection of dictionaries \mathcal{D}_s independent of the letter probabilities P such that the compression $\max_s \log |\mathcal{D}_s| / E(L)$ is close to the entropy of the source. We note that this definition is in general different from the definition of compression rate in Ziv (1990). The two are equivalent when the resultant segment source is ergodic. If the underlying source is ergodic we can ensure the segment source is ergodic by adding a constant number of code words to the dictionary. Thus the definitions are equivalent in studying the asymptotic rates of the code.

3. A CODING METHOD

In this section we will give a method to achieve optimal coding for a Markov source. As mentioned earlier we assume that the encoder and the decoder know the next-state function though the letter probability function is unknown.

We first describe a method to form a complete, prefix free dictionary given a distribution P . We assume that P satisfies $\sum_{x \in A} P(y^*x) = P(y^*)$ and $\sum_{x \in A} P(x) = 1$. Define the dictionary \mathcal{D} by $x^* \in \mathcal{D}$ if $P(x^*) \leq c$ and no prefix of x^* satisfies this property. Clearly \mathcal{D} is prefix free, it is also complete if the distribution is such that $P(x^n)$ goes to zero as n goes to infinity for any infinite sequence x .

We will create dictionaries using the method described above for a finite but growing set of possible letter probability matrices and combine all these dictionaries to form a dictionary which will work well for any possible letter probability matrix. The method we use for combining dictionaries was given by Savari and Gallager (1997). Consider k column vectors of integers $\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k$, each vector of size a . The vectors are chosen so that each entry of a vector is between 1 and m and the entries in a particular vector sum to $m+1$. Thus $\mathbf{I}_j/(m+1)$ defines a distribution on A for each j . Let $Q_{\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k} = [\mathbf{I}_1 \mathbf{I}_2 \dots \mathbf{I}_k] / (m+1)$. Then Q is a valid letter probability matrix for each set of vectors chosen according to the restrictions above. Each valid Q defines along with an initial state s defines a distribution $Q_{\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k}(\cdot | s)$. We will form $\mathcal{D}_s(\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k)$ by using the procedure described above with

$$c = \frac{m^{k(a-1)}(m+1)}{M \min(i_1, i_2, \dots, i_k)}$$

where i_j is the minimum entry in vector \mathbf{I}_j .

We note that because of the way the dictionary is constructed we have

$$Q_{\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k}(x^* | s) \geq c \frac{\min(i_1, i_2, \dots, i_k)}{m+1}$$

for $x^* \in \mathcal{D}_s(\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k)$. Thus since

$$\sum_{x^* \in \mathcal{D}_s(\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k)} Q_{\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k}(x^* | s) = 1$$

we have

$$|\mathcal{D}_s(\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k)| \leq \frac{m+1}{c \min(i_1, i_2, \dots, i_k)}$$

Substituting in the value of c we have used to form the dictionary we have

$$|\mathcal{D}_s(\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k)| \leq \frac{M}{m^{k(a-1)}}. \quad (1)$$

We have so far described how to form the dictionary for a particular letter probability matrix Q depending on $\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k$. We can thus obtain dictionaries for each of the different possible valid vectors $\mathbf{I}_1, \mathbf{I}_2, \dots, \mathbf{I}_k$. We will combine these dictionaries by taking a union of all the dictionaries and then removing elements so as to make the union prefix free. Let the dictionary so obtained be \mathcal{D}_s .

The compression achieved by this scheme with $m = \sqrt{\log M}$ and when N results in a irreducible, aperiodic source is given by the following theorem.

THEOREM 1. *For any $\delta \geq 0$ and for any letter probability function P with $0 < P(x|s) < 1$ for each $x \in A, s \in S$, there exists $M(P, \delta) < \infty$ such that*

$$R = \frac{\max_s \log |\mathcal{D}_s|}{E(L)} \leq H(P) \left(1 + (1 + \delta) \frac{k(a-1) \log \log M}{2 \log M} \right)$$

for all $M \geq M(P, \delta)$.

We have constructed the dictionary by a combining dictionaries for a finite set of parameters in the parameter space. The closer the true parameter is to a parameter within the finite set the better the coding performance. This gain is to be traded off against the loss occurring due to combining many dictionaries to form one dictionary. It can be shown that if we take $m = \sqrt{\log M}$ we achieve a good tradeoff between these two terms. Details of the proof can be found in Visweswariah, Kulkarni and Verdú (1999).

4. CONVERSE

In this section we will give a lower bound on the performance achievable by any code. We will assume that the next-state function is such that the states are irreducible. We will also represent the letter probability matrix P as a vector \mathbf{p} with $k(a-1)$ components with $p_{(i-1)(a-1)+j} = P(j|s=i)$, $i = 1, 2, \dots, k$ and $j = 1, 2, \dots, a-1$. Note that we do not need ka components because of the constraint that the rows of P must sum to 1. The converse will not be for every parameter value \mathbf{p} but rather for 'most' parameter values. The converse result is stated precisely in the theorem below. The proof is a modification of the converse for i.i.e. sources given in Tjalkens and Willems (1992).

THEOREM 2. For all $\delta > 0$ and any variable-to-fixed length code with M large enough

$$R_p \geq H(P) \left(1 + (1 - \delta) \frac{k(a-1) \log \log M}{2 \log M} \right)$$

for all valid $p \in [0, 1]^{k(a-1)}$ except for those in a set whose volume goes to zero as M increases.

We give an outline of the proof. Let B be the set of parameters for which the rate R is 'small'. Consider the set $C \in B$ of maximal size such that every pair of parameters in C are ε -apart. Let the size of C be N . We can upper bound the volume of B by upper bounding N . To bound N we show that if a code is 'good' for a parameter then the probability of a leaf occurring which is typical with that parameter is high. We can show that this requires the number of leaves typical with that parameter to be high. Since the total number of leaves is limited by M this limits the N . We can then show that the volume of B vanishes asymptotically. The details of the proof can be found in Visweswariah, Kulkarni and Verdú (1999).

5. A PRACTICAL CODING METHOD

We describe and investigate the performance of a coding method described in Nissenbaum and Feder (1994). The method gives a variable-to-fixed length source coding method which can be used in conjunction with a model for the source statistics which provides estimates of probabilities based on the past data. This scheme can be used to obtain a variable-to-fixed length coder for Markov sources and in general for stationary ergodic sources. The coding method is closely related to arithmetic coding. The method described in Section 3 is optimal but is not a method which would be implementable in practice unlike the the coding method in Nissenbaum and Feder (1994).

We will first describe the method for variable-to-fixed length coding in detail. We will assume that our input is from a source X that takes values on a binary alphabet $A = \{0, 1\}$. We assume a binary alphabet for convenience and the method described works for any finite alphabet. Let $\hat{P} : A^* \rightarrow [0, 1]$ be a function which satisfies the following consistency conditions: $\hat{P}(0) + \hat{P}(1) = 1$ and $\hat{P}(x0) + \hat{P}(x1) = \hat{P}(x)$ for any $x \in A^*$. Now we will use the probabilities that \hat{P} provides to code the source. Suppose that we want to build a coding tree with M leaves. Initially the range for our index is $[1, M]$. Suppose that the source output is a string $x = x_1, x_2, x_3, \dots$. Suppose that after stage m our index range is $[i_l(m), i_h(m)]$. If

$x_{m+1} = 0$ we change keep the lower end of the index range the same and we change the higher end of the range to $i_l(m) + \max(0, \lfloor (i_h(m) - i_l(m)) * \hat{P}(0 | x_1^m) \rfloor - 1)$. If $x(m+1) = 1$ we keep the high end of the index range fixed and we change the lower end to $i_l(m) + \max(1, \lfloor (i_h(m) - i_l(m)) * \hat{P}(0 | x_1^m) \rfloor)$. We proceed in this way until there is only one number in the index range. This number is the codeword for the input string and since we started with a range of $[1, M]$ we can code each output index with $\lceil \log M \rceil$ bits. If $\hat{P}(x_1^l)$ goes to zero as l increases for each infinite string x then the method described above always terminates and thus describes a tree with M leaves and can be used for variable-to-fixed length coding. Note that we need not build and store the tree, we can compute the parsing and the output index if $\hat{P}(0 | x)$ can be computed for each string $x \in A^*$. The decoder must have access to the same \hat{P} and then the coding process of the encoder can be reversed at the decoding end.

We investigate the performance of the coding method for Markov sources. The performance bound that we show is not optimal but is close to being optimal. To determine the performance of the coding method we will need to upper bound the probability of the leaves of the coding tree. Consider a string x_1^l which is a leaf of the coding tree described above. Suppose the number of possible indices left after stage m is $r(m)$ then the number of indices left after seeing x_{m+1} is at least $r(m) * \hat{P}(x_{m+1} | x_1^m) - 1$. Since x_1^l is a leaf we know that after l stages there is only one index left. Thus we have that $r(l-1) * \hat{P}(x_l | x_1^{l-1}) - 1 \leq 1$. Proceeding backward we have that

$$(\dots(((M\mathbf{p}(x_1) - 1)\mathbf{p}(x_2 | x_1) - 1)\mathbf{p}(x_3 | x_1^2) - 1) \dots)\mathbf{p}(x_l | x_1^{l-1}) - 1 \leq 1.$$

Equivalently we have

$$M\mathbf{p}(x_1^l) \leq 2 + \sum_{j=1}^{l-1} \mathbf{p}(x_{j+1}^l | x_1^j). \tag{2}$$

Suppose we know that our source is a Markov source with memory at most D . We use the Context-tree weighing method described in Willems, Shtarkov and Tjalkens (1995) to obtain the \mathbf{p} required to code the source. This method gives us a coding distribution $\mathbf{p}(\cdot | x_{-D}^{-1})$ depending on the past D bits. Thus the coding method will utilize 2^D coding trees choosing one based on the past D bits.

Now using Equations (39) and (40) from Tjalkens and Willems (1987) we have

$$\sum_r P(G_r) \sum_{s \in G_r} q(s | G_r) \sum_{x^* \in D_s} P(x^* | s) \log \frac{1}{P(x^* | s)} = E(L)H. \tag{3}$$

In the above equation $E(L)$ is the expected phrase length, H is the entropy rate of the source. G_r are the irreducible sets that are induced by the 2^D coding trees. $P(G_r)$ is the probability that the source will be in a state in G_r and $q(s | G_r)$ is the probability the the source is in state $s \in G_r$ given that the source-state is in G_r . Note that $\cup G_r = \{0, 1\}^D$. Since

$$\begin{aligned} \sum_{x^* \in D_s} P(x^* | s) \log \frac{1}{P(x^* | s)} &= \sum_{x^* \in D_s} P(x^* | s) \log \frac{p(x^* | s)}{P(x^* | s)} \\ &+ \sum_{x^* \in D_s} P(x^* | s) \log \frac{1}{p(x^* | s)} \end{aligned} \tag{4}$$

we can lower bound $E(L)$ by lower bounding the two terms on the R.H.S. of Equation 4. Now from Theorem 2 in Willems, Shtarkov and Tjalkens (1995) we have

$$\begin{aligned} \sum_{x^* \in D_s} P(x^* | s) \log \frac{p(x^* | s)}{P(x^* | s)} &\geq - \sum_{x^* \in D_s} P(x^* | s) (k\gamma \left(\frac{l(x^*)}{k} \right) + 2k - 1) \\ &\geq -(k\gamma \left(\frac{E(L | s)}{k} \right) + 2k - 1) \end{aligned}$$

where k is the number of parameters in the unknown source and

$$\gamma(z) = \begin{cases} z, & \text{for } 0 \leq z < 1 \\ \frac{1}{2} \log z + 1, & \text{for } z \geq 1 \end{cases}$$

The second inequality above follows from the fact that $-\gamma$ is convex.

Consider now the second term. We have

$$\begin{aligned} \sum_{x^* \in D_s} P(x^* | s) \log \frac{1}{p(x^* | s)} &\geq \sum_{x^* \in D_s} P(x^* | s) \log \frac{M}{2 + \sum_{j=1}^{l(x^*)-1} p(x_{j+1}^* | x_j^*)} \\ &\geq \log M - \sum_{x^* \in D_s} P(x^* | s) \log (l(x^*) + 1) \\ &\geq \log M - \log(E(L | s) + 1). \end{aligned}$$

The first inequality follows from Equation 2, the second because $P(x_{j+1}^* | x_j^*) \leq 1$ and the third from the convexity of $-\log(\cdot)$.

Combining these two lower bound with Equation 3 we have,

$$\begin{aligned} E(L)H &\geq \sum_r P(G_r) \sum_{s \in G_r} q(s | G_r) - (k\gamma \left(\frac{E(L | s)}{k} \right) + 2k - 1) + \log M - \log(E(L | s) + 1) \\ &\geq -(k\gamma \left(\frac{E(L)}{k} \right) + 2k - 1) + \log M - \log(E(L) + 1) \end{aligned}$$

where the second inequality follows from the definition of $E(L)$ and the convexity of $-\gamma$ and $-\log(\cdot)$ functions.

Using Equation 3 and the fact that entropy of a distribution on an alphabet of size M is no larger than $\log M$ we have that $HE(L) \leq \log M$. Thus we have from the monotonicity of $\log(\cdot)$ and $\gamma(\cdot)$

$$E(L)H \geq -\left(k\gamma\left(\frac{\log M}{Hk}\right) + 2k - 1\right) + \log M - \log(\log M/H + 1).$$

Equivalently the compression ratio $R = \log M/E(L)$ is upper bounded as

$$R \leq \frac{\log M}{\log M - \log\left(\frac{\log M}{H} + 1\right) - \left(k\gamma\left(\frac{\log M}{Hk}\right) + 2k - 1\right)} H.$$

Since $\gamma(z) \leq 1 + (\log z)/2$ we have for any $\delta > 0$ we have for sufficiently large M depending on the source and δ ,

$$R \leq (1 + (1 + \delta) \frac{k \log \log M}{2 \log M}) H.$$

The redundancy we have shown is of the order of $((k + 2) \log \log M)/2 \log M$ while the optimal redundancy is $(k \log \log M)/2 \log M$, thus the extra redundancy is $(\log \log M)/\log M$.

REFERENCES

- [1] B. Nissenbaum and M. Feder (1994), "An adaptive variable-to-fixed lossless coding scheme", Proc. IEEE International Symposium on Information Theory, Trondheim, p. 193.
- [2] S.A. Savari and R.G. Gallager (1997), "Generalized Tunstall codes for sources with memory", *IEEE Trans. Information Theory*, **43**, 658-668.
- [3] T.J. Tjalkens and F.M.J. Willems (1987), "Variable to fixed-length codes for Markov sources", *IEEE Trans. Information Theory* **33**, 246-257.
- [4] ——— (1992), "Universal variable-to-fixed-length source codes based on Lawrence's algorithm", *IEEE Trans. Information Theory*, **38**, 247-253.
- [5] B.P. Tunstall (1967), Synthesis of Noiseless Compression Codes, Ph.D. Thesis, Georgia Inst. of Technology., Atlanta, GA.
- [6] K. Visweswariah, S. Kulkarni and S. Verdú (1999), "Universal variable-to-fixed length source codes" (submitted *IEEE Trans. Information Theory*).
- [7] F.M.J. Willems, Y.M. Shtarkov and T.J. Tjalkens (1995), "The context-tree weighting method: Basic properties", *IEEE Trans. Information Theory*, **44**, 653-664.
- [8] J. Ziv (1990), "Variable-to-fixed length codes are better than fixed-to-variable length codes for Markov sources", *IEEE Trans. Information Theory*, **36**, 861-863.