Electronics and Communications in Japan, Part 1, Vol 71, No. 2, 1988
Translated from Denshi Tsushin Gakkai Ronbunshi, Vol. 69-A, No. 8, August 1986, pp. 974-982

Modeling by MDL Criterion for Adaptive Data Compression

Hidetoshi Yokoo, Member

Faculty of Engineering, Yamagata University, Yonezawa, Japan 992

SUMMARY

This paper proposes a high-performance noiseless data compression model which can compress strings generated by a binary Markov information source without prior knowledge of the source. The adaptive expression of the existing universal codes is described by a model with separated sampler and encoder, and the learning structure of each code and the encoding process are described. Through such a discussion, the intuitive reasons are clarified as to why the codes, despite their asymptotic optimality, do not compress well at the start, pointing out room for improvement. Then, the MDL criterion is introduced into each encoder, to improve the performance, while keeping the sampler fixed. The MDL criterion is an estimation for the minimum code length, including the representation cost of the parameters. Based on this criterion, the Markov order which is to be assumed for each symbol is adaptively determined. Since the same determination can be made at the time of decoding, the unique decodability is guaranteed. In the second half of this paper, more detailed techniques are discussed, and a remarkable improvement is demonstrated by a computer simulation. It is also shown that the sampler of the proposed universal code is of a high-performance, realizing two kinds of encoders.

1. Introduction

At present, various kinds of data are processed daily on a large scale, and it is desired to establish a compression method without any prior knowledge of the data. The data compression may or may not anticipate a certain distortion in the decoding. The technique considered in this paper is the distortionless (noiseless) compression of the discrete data.

The basic methods of the data compression without prior knowledge can be classified into two main groups. The first, the two-pass method, involves a preliminary pass to acquire the model parameters, and a second pass for actual encoding. In the second method, called adaptive coding, the parameter gathering and encoding are done in a single pass through the data. Among the methods which employ the Huffman code [1] are the conventional two-pass method by Pechura [2], and the adaptive method by Gallager [3] and Vitter [4]. In these methods, however, the assumed model is the symbolwise independent and identical distribution or its slight extension. Consequently, the range of their effective application is limited. In the usual image data compression, the current pixel is encoded using the predetermined neighboring pixels. A data compression method which can determine whether or not the neighboring pixels are really necessary has not been reported.

The Ziv-Lempel compression method [5], based on their incremental algorithm, which is asymptotically optimum for infinite string generated by a stationary ergodic source, is expected to have a capability to identify the underlying structure of the string. The method, however, has several problems in practical application, and some improvements have been proposed [6 - 8]. The most essential problem is that the method does not estimate explicitly the structure, and even if the parameters are estimated with a high accuracy, the method cannot make that determination.

In both two-pass and adaptive methods, the compressed output code contains the description of the data properties, in addition to the data itself. If the model describing the data has a higher order, the data itself can more efficiently be described, by sacrificing an increase in the parameter representation cost. For this problem, Rissanen [9, 10] presented the general expression for the total code length. The minimum of that expression is called MDL (minimum description length), which he used as the criterion in the comprehensive treatment of the prediction and estimation, with

ISSN8756-6621/88/0002-0001$7.50/0

the common link to the information in the data. In the course of that discussion, he proposed an adaptive data compression, which led to the solution [11].

This paper considers several universal codes including the Ziv-Lempel code, and presents a model with separate units for the parameter learning (sampler) and for the encoding (encoder). It is shown that by introducing the MDL criterion into the encoder, a high-performance data compression model can be realized for a binary Markov source with unknown order and unknown parameters. Especially, for the sampler of the universal code proposed by the present author [12], there exist both the encoder with a remarkable universality and the encoder with the high-performance in the foregoing sense. A similarity is pointed out between the data compression method by the latter model and by Rissanen's method [11]. As a result, the significance of Rissanen's method is discussed, which has been regarded as a means of proof of a theorem rather than the compression algorithm.

The performance discussed in this paper is that as a model,and the realization method, e.g., arithmetic coding [13 - 15], is not discussed. Consequently, the compression performance is compared in terms of the entropy achievable by each model. In the following $\log_2 a$ is written as $\log a$.

2. Modeling of Adaptive Data Compression

As the source of data which are the object of compression, consider a stationary Markov information source with a finite order and with alphabet $\mathcal{X} = \{0, 1\}$. The order and the true values of parameters are assumed as unknown. Letting an element of $\mathcal{X}$ be x, the other $1 - x$ is denoted by $\bar{x}$. A sequence composed of elements of $\mathcal{X}$ is denoted by s. Usually, a sequence is represented by its length and the output at each time as

$$x(1, n) = x_1 x_2 \cdots x_t \cdots x_n ,$$
$$x_t \in \mathcal{X} \qquad 1 \leqq t \leqq n$$

The empty sequence with length 0 is written as λ. It is defined that $x_0 = \lambda$ and $x(i, j) = \lambda$ for $j < i$.

The output alphabet, which is the output of the data compressor, is written as $\mathcal{Y} = \{0, 1\}$. The output sequence for the input $x(1, n)$ is written as

$$y(1, m) = y_1 y_2 \cdots y_t \cdots y_m ,$$

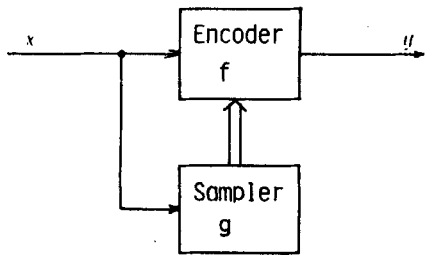


Fig. 1. Data compression system.

$$y_t \in \mathcal{Y} \qquad 1 \leqq t \leqq m$$

The compression ratio is given by m/n.

For x_t, the immediately preceding sequence $z = x(t - i, t - 1)$ of length i ($i \geqq 0$), including λ, is called context [11, 16] (or class [16]). Its length i is denoted by $|z|$. By the assumption for the source, there exists an M for sufficiently large t such that x_t has the conditional probability distribution

$$P(x \mid x(t-i, t-1)) = P(x \mid x(t-M, t-1))$$

for any $i \geqq M$. The context x $(t - M, t - 1)$ with the minimum length realizing this distribution is called the true context of x_t.

For any context z, the probability of context $P(z)$ can be defined, satisfying the condition

$$P(\lambda) = 1 ,$$
$$P(z) = \sum_{x \in \mathcal{X}} P(zx) \tag{1}$$

The stationarity of the source is represented as [16]

$$P(z) = \sum_{x \in \mathcal{X}} P(xz) \tag{2}$$

The model for the adaptive data compressor is essentially the same as that of Rissanen and Langdon [16]. The feature of the model in this paper is that the encoder and the sampler are separated as follows. As shown in Fig. 1, the encoder encodes the input x_t with the knowledge of the state of the sampler, and then performs the state transition with the knowledge of x_t.

The encoder has the structure function [16] f to determine the context z_t for encoding x_t. The sampler has the structure

function g to determine the set of contexts with x_t as the sample. The encoder calculates the conditional probability of $x_t = x$ for $x(0, t-1)$ by

$$P(x|x(0, t-1)) = P(x|z_t, x_0^{t-1}[z_t]) \quad (3)$$

where $x_0^{t-1}[z_t]$ is a sequence constructed as follows. For $z_t = f(x(0, t-1))$, the set of symbols immediately after the contexts, selected by the sampler from the sequence $x(0, t-1)$ as being equal to

$$\{x_i | z_t \in g(x(0, i-1)), \quad 1 \leqq i \leqq t-1\}$$

are concatenated successively.

Thus, $g(x(0, t-1))$ is the function specifying the set of contexts. Letting its size be $|g(x(0, t-1))|$,

$$\sum_{t=1}^{n} |g(x(0, t-1))|$$

is called the gross number of samples or total learning [12]. For the sequence $x(1, 12) = 001011010010$, for example, the number of symbol occurrences $c(\cdot)$, and the number of occurrences $c(\cdot|x)$ immediately after x are counted as follows:

$$c(0)=7, \quad c(1)=5, \quad c(0|0)=2,$$
$$c(1|0)=4, \quad c(1|0)=4, \text{ and } c(1|1)=1$$

Then, the function g is represented as

$$g(x(0, t-1)) = \begin{cases} \{\lambda\} & t = 1, \\ \{\lambda, x(t-1, t-1)\} & 2 \leqq t \leqq 12 \end{cases}$$

The gross number of samples is 23.

The ideal code length [11, 16] of the model for sequence $x(1, n)$ is given by

$$I(x(1, n)) \triangleq -\log P(x(1, n)) = -\sum_{t=1}^{n} \log P(x_t | x(0, t-1)) \quad (4)$$

Thus, the construction of the adaptive coding model reduces to the determination of the structure functions f, g and the computation algorithm of Eq. (3).

Once the model is determined, it can be realized by the arithmetic code, and the compression ratio can be as close to the limit

$$\rho(x(1, n)) \triangleq \frac{I(x(1, n))}{n} \quad (5)$$

as desired. Consequently, the compression performance is compared by Eq. (5). It is called the compression ratio by ideal code length, or the entropy of the model. Three examples of the adaptive data compression model are presented in the following.

Example 1. Consider the case where $f(s) = \lambda$ and $g(s) = \{\lambda\}$ for any sequence s. In this case, $x_0^{t-1}[z_t] = x(0, t-1)$ holds for $1 \leqq t \leqq n$. Let

$c(0)$ = {number of 0's in $(0, t-1)$} + 1,
$c(1)$ = {number of 1's in $(0, t-1)$} + 1

Let Eq. (3) be

$$P(x|x(0, t-1)) = P(x|\lambda, x(0, t-1)) = \frac{c(x)}{c(0)+c(1)}$$

Then, the ideal code length is given by

$$I(x(1, n)) = \log \binom{n}{m_n} + \log(n+1) \quad (6)$$

where m_n is the number of 0's in $x(1, n)$.

This model is known [18, 19] as the adaptive representation of Schalkwijk code [17]. Letting

$$q = \lim_{n \to \infty} \frac{m_n}{n}$$

there applies

$$\lim_{n \to \infty} \rho(x(1, n)) = \mathcal{H}(q) \triangleq -q \log q - (1-q) \log(1-q)$$

indicating that the model is asymptotically optimal for the memoryless ergodic source.

Example 2 (model 2).

This is the model which calculates Eq. (3), constructing a tree, by representing the context by a node and attaching the value $(c(0|z), c(1|z))$ to node z. As the initial tree, consider Fig. 2, where the root has the value (1, 1), which is followed by the branches corresponding to the source symbols 0 and 1, respectively, together with their leaves. The pointer is placed at the root. In the following, the node of the tree with already-given value is defined as the internal node, and those without the given

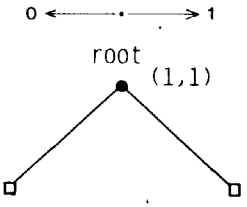


Fig. 2. Initial state of model 2 and model 3.

value are defined as the leaf. The sequence corresponding to the path from the root to each node is defined as the context.

For the input x_t, the encoder determines

$$P(x \mid z_t,\ x_0^{t-1}[z_t]) = \frac{c(x \mid z_t)}{c(0 \mid z_t) + c(1 \mid z_t)}$$

from the value $(c(0|z_t),\ c(1|z_t))$ at the pointer node z_t. The sampler updates the value at that stage as

$$c(x_t \mid z_t) := c(x_t \mid z_t) + 1$$

and shifts the pointer to the forward node of the branch corresponding to x_t. If that node is an internal node, the processing for x_t is completed. If it is a leaf, the value (1, 1) is attached to that node, and two branches and leaves are continued from that node. Then the pointer is placed at the root, and the processing for x_t is completed.

By repeating the forementioned procedure, an ideal code length is arrived at [11, 20] by eliminating the trivial redundancies from the Ziv-Lempel method [5]. This code is asymptotically optimum in the following sense. When the information source is a stationary ergodic source with entropy H,

$$Pr[\rho(s) = H] = 1$$

is achieved for the infinite sequence s generated by the source.

In this model,

$$g(x(0,\ t-1)) = \{f(x(0,\ t-1))\},\quad t \geqq 1$$

holds for any sequence $x(0,\ t-1)$. Consequently, the gross number of samples for $x(1,\ n)$ is n. On the other hand, the value $c(x|z)$ at each node is the number of occurrences of symbol x immediately after the context z, as counted by the sampler according to the structure function g, plus 1.

Consequently, if the input sequence is generated by a memoryless ergodic source,

$$p(x \mid z) \triangleq \frac{c(x \mid z)}{c(0 \mid z) + c(1 \mid z)} \qquad (7)$$

approaches asymptotically the true probability of symbol x in any context z. If the information source has a memory, Eq. (7) approaches asymptotically the probability of x immediately after the true context, at a later stage.

Example 3 (model 3).

In this example, the encoder operates in entirely the same way as in example 2. On the other hand, the sampler has a buffer to store the subsequence, in addition to a tree similar to that in example 2. As a result, both f and g are different from those in example 2. The subsequence of length K in the buffer, which is needed in the computation, is represented as

$$b(0,\ K) = b_0 b_1 b_2 \cdots b_K\ ,$$
$$b_0 = \lambda$$

The initial state is defined as empty, i.e., $K = 0$. The initial state of the tree is defined in the same way as in example 2. The pointer is placed at the root.

For input x_t, the encoder determines

$$P(x \mid z_t,\ x_0^{t-1}[z_t]) = \frac{c(x \mid z_t)}{c(0 \mid z_t) + c(1 \mid z_t)}$$

from the value at the pointer z_t, and performs the encoding. On the other hand, the sampler sets that

$$K := K + 1,$$
$$b_K := x_t$$

After storing x_t in the buffer, it places the pointer at the forward node of the branch corresponding to x_t. If that node is an internal node, the processing for x_t is completed. If it is a leaf, the tree and the content of the buffer are updated by the following procedure. Then the pointer is placed at the root, and the processing for x_t is completed.

Procedure Update.

(1) $i := 1$.

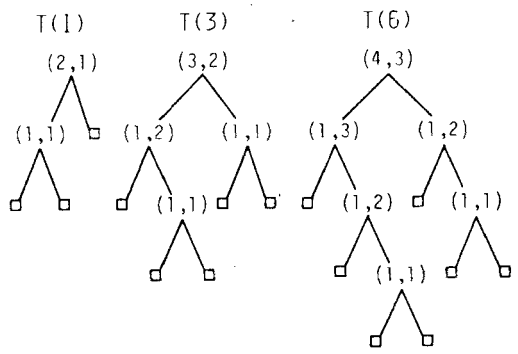


Fig. 3. Examples of tree representation.

(2) For k such that $i \leqq k \leqq K$, the following procedure is repeated as long as there exists a path from the root to the leaf corresponding to the subsequence $b(i, k)$.

(2.1) For all j such that $i \leqq j \leqq K$, let

$$c(b_j | b(i, j-1)) := c(b_j | b(i, j-1)) + 1.$$

(2.2) Attach the value (1, 1) to the leaf $b(i, k)$, and attach two branches with new leaves.

(2.3) $i := i + 1$.

(3) $b(0, K-i+1) := b_0 b(i, K)$.

(4) $K := K - i + 1$.

By repeating the foregoing procedure, the universal code [12] proposed by the author is obtained, and is shown to have the same asymptotic optimality as in example 2. Figure 3 shows several examples for the sequence $x(1, 6) = 001011$, where the tree after encoding x_t is defined as $T(t)$.

In either of the models in example 2 or 3, the maximum depth of the tree at time t is $O(\log t)$. In the foregoing model, the counting of $c(x|z)$ in each x_t is concerned with the same order of nodes proportional to the depth of the tree, and the gross number of samples for $x(1, n)$ is of the order of $O(n \log n)$ [12]. (End of Example.)

The three foregoing models have asymptotic optimality, but the convergence is very slow. If the original data do not have a sufficient length, then the data may expand. As a result, those models have a starting characteristic with a large aftereffect. Consequently, they are not useful for the actual data from the viewpoint of the compression ratio.

3. Improvement of Model by MDL Criterion

3.1 Formulation of problem

It is not easy to design a model with a high compression performance, considering the whole part of the model for the data compression. Consequently, in the following, the structure function g of the sampler is fixed, and the structure function f of the encoder and the computation algorithm for Eq. (3) are determined so that a high compression performance is achieved.

When, for example, an expansion of the data is anticipated at the start of encoding, it is better to produce x_t directly as the output. In the case of example 1, the entropy $\rho(x(1, t))$ of the model is estimated. By comparing the result with 1, the code for x_t should be determined. To indicate clearly the relation to the later discussion, the following approximation is considered.

When the number of 0's in $x(0, t-1)$ is m_{t-1}, the probability of 0 is estimated as

$$p_t = \frac{m_{t-1} + 1}{t + 1}$$

Then, the entropy of the model for $x(1, t)$ is approximated asymptotically as

$$\rho(x(1, t)) \simeq \rho_1(t) \tag{8}$$

$$\rho_1(t) \triangleq \mathcal{H}(p_t) + \frac{1}{2t} \log t. \tag{9}$$

Letting

$$\rho_0(t) \equiv 1$$

the encoding of x_t can be made by the model

$$P(0|x(0, t-1)) = \begin{cases} \frac{1}{2} & \rho_0(t) \leqq \rho_1(t) \\ p_t & \rho_0(t) > \rho_1(t) \end{cases} \tag{10}$$
$$P(1|x(0, t-1)) = 1 - P(0|x(0, t-1))$$

In the cases of examples 2 and 3, such an improvement is not obvious. Consequently, for examples 2 and 3, a higher performance encoder is designed for the given structure function g of the sampler.

3.2 MDL criterion and its application

The fact that the structure function of the sampler is fixed implies that several

statistical parameters are already obtained concerning $x(0, t)$ in the encoding of x_t. If as many parameters as possible are utilized, the length of the code itself can be decreased, while the representation cost for the parameters is increased. if the representation cost for the parameters is suppressed, the length of the code itself is increased. Consequently, an optimization has to be made considering both aspects.

For this problem, Rissanen [9, 10] presented a general proof for the following fact. If k independent parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_k)$ are chosen, Eq. (4) can be approximated by

$$I_k(x(1, n)) = -\log P_{\boldsymbol{\theta}}(x(1, n)) + \frac{1}{2} k \log n \qquad (11)$$

where $P_{\theta}(x(1, n))$ is the likelihood of $x(1, n)$ for those parameters. The interpretation is that the second term corresponds to the representation cost for the parameters in the two-scanning method, and to the cumulative estimation error in the adaptive method. From such a viewpoint, the encoding should be made using the parameters achieving $\min_{k,\boldsymbol{\theta}}\{I_k(x(1, n))\}$. This is called MDL criterion.

In this paper, the adaptive encoding is performed using the MDL criterion as follows. From the parameters $\{\boldsymbol{\theta}\}$ which can be estimated from $x(0, t-1)$, k and $\boldsymbol{\theta}$ are selected so that

$$\frac{I_k(x(0,t))}{t} = \frac{-\log P_{\boldsymbol{\theta}}(x(0,t))}{t} + \frac{k \log t}{2t} \qquad (12)$$

is minimized, and the encoding is performed. The probability of context z and the probability of symbol x in context z, estimated from $x(0, t-1)$, are denoted by $p(z)$ and $p(x\,z)$, respectively, in the encoding of x_t.

Letting $k = 0$, for example, Eq. (12) gives 1. For $k = 1$, it is

$$\frac{I_1(x(0, t))}{t} = \mathcal{H}(p(0 \mid \lambda)) + \frac{\log t}{2t} = \mathcal{H}(p(0)) + \frac{\log t}{2t}$$

In example 1, the number of independent parameters, which can be estimated by the sampler, is at most 1. In fact, the model based on Eq. (10) minimizes Eq. (12) for $k = 0$ and 1. In more general cases, however, it is not the number of parameters but the context itself, which can be controlled in the encoding of x_t. Equation (12) should be considered from such a viewpoint.

Let the number of parameters among k parameters, which concern the encoding of x_t, be d_t. Then by the Markovian property of the information source,

$$d_t = \begin{cases} 0 & y_t = x_t, \\ |z_t| + 1 & z_t = f(x(0, t-1)) \end{cases} \qquad (13)$$

This is called the local order of x_t. The value of d_t for the true context is called the true local order. In general, $0 \leqq d_t \leqq k$. For $k = 0$ and 1, $d_t = k$ by definition.

By contrast, consider, for example, $d_t = 2$, i.e., $z_t = x(t-1, t-1) = x_{t-1}$. Those in $x(0, t-1)$ whose context ends up with $\bar{x}_{t-1}$ is out of the control in encoding x_t. Representing their contributions by C, one can write that

$$I_k(x(0, t)) = t\,p(x_{t-1})\mathcal{H}(p(0 \mid x_{t-1})) + t\,p(\bar{x}_{t-1})C + \frac{1}{2} k \log t$$

Increasing k by 1 by increasing d_t by 1,

$$I_{k+1}(x(0, t)) = t\,p(x(t-2, t-1))\mathcal{H}(p(0 \mid x(t-2, t-1))) + t\,p(\bar{x}_{t-2}x_{t-1})\mathcal{H}(p(0 \mid \bar{x}_{t-2}x_{t-1})) + t\,p(\bar{x}_{t-1})C + \frac{1}{2}(k+1)\log t$$

The function to minimize is

$$\rho_d(t) \triangleq H_d(t) + \frac{d}{2}\,\frac{\log t}{t} \qquad (14)$$

$$H_0(t) = 1$$

$$H_1(t) = \mathcal{H}(p(0))$$

$$H_d(t) = H_{d-1}(t) - p(x(t-d+2, t-1)) \cdot \mathcal{H}(p(0 \mid x(t-d+2, t-1))) + p(x(t-d+1, t-1)) \cdot \mathcal{H}(p(0 \mid x(t-d+1, t-1))) + p(\bar{x}_{t-d+1}x(t-d+2, t-1)) \cdot \mathcal{H}(p(0 \mid \bar{x}_{t-d+1}x(t-d+2, t-1))) \qquad d \geqq 2$$

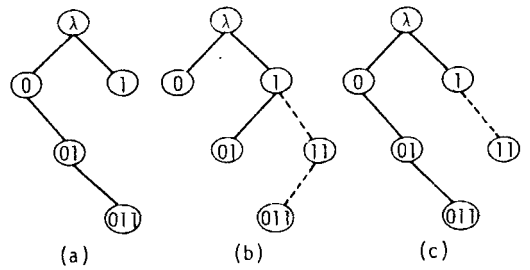


Fig. 4. Transformation of tree structure.

Thus, it suffices to determine $d_t = d$ minimizing Eq. (14) in the range of estimation, and to encode x_t by the model

$$P(x|x(0,\ t-1)) = \begin{cases} \dfrac{1}{2} & d_t = 0 \\ p(x|x(t-d_t+1,\ t-1)) & d_t \geqq 1 \end{cases} \tag{15}$$

3.3 Transformation of tree structure

In examples 2 and 3, $c(x|z)$ is the number of occurrence of symbol x in the context z as counted by the structure function g of the sampler, plus 1. From this, $p(z)$ and $p(x|z)$ can be estimated. In the minimization by Eq. (15), the candidates of the context following z to be examined are $0z$ and $1z$, while the child nodes following the context z on the given tree structure are $z0$ and $z1$. This makes the retrieval difficult.

Consider the case in example 3, for example, where x_7 is to be encoded after $x(1, 6)$. Among the candidates λ, $x(6, 6) = 1$, $x(5, 6) = 11$ and $x(4, 6) = 011$ of the context, $x(4, 6)$ exists on a path different from the other three, which makes the retrieval time-consuming. To make the retrieval easy, the data structure is transformed to produce a dual tree, where $0z$ and $1z$ are child nodes of the node z. If an arbitrary tree is given, not only is this transformation of data structure time-consuming, but also the cases may exist where it is impossible, as is shown by an example of Fig. 4.

In Fig. 4, the node corresponding to 011 in the original tree in (a) must be a child node after transformation, which is connected by the branch 0 to the node corresponding to 11. The node corresponding to 11, however, does not yet exist. Consequently, even if the node does not exist, the node which is needed in the transformation is added, being called a virtual node. After the transformation [Fig. 4(b)], the virtual node is also added to the original tree [Fig. 4(c)]. By the transformation of the tree structure, the tree grows up each time by adding a node, and the compression is performed without changing the order of the computation time.

In other words, as was discussed earlier, the maximum depth of the original tree corresponding to the sequence of total length n is $O(\log n)$, and the depth of the virtual node does not exceed that value. Consequently, the computational complexity required in the generation of all nodes, including the virtual nodes, is $O(n \log n)$. On the other hand, the context of depth up to $O(\log t)$ is examined in the encoding of x_t, and the total computational complexity is $O(n \log n)$.

3.4 Coding algorithm

Based on the preceding preparations, the encoding algorithm is constructed for example 3. In example 3, $c(x|\lambda)$ for the context λ is counted for all symbols in $x(0, t - 1)$. Letting the estimated value be $p(x|\lambda)$, one can write that

$$p(x|\lambda) = \frac{c(x|\lambda)}{c(0|\lambda) + c(1|\lambda)} \tag{16}$$

For the more general context,

$$p(x|z) = \frac{c(x|z)}{c(0|z) + c(1|z)} \tag{17}$$

is used.

Defining that

$$c(z) \triangleq c(0|z) + c(1|z)$$

the following estimation $p(z)$ can be used.

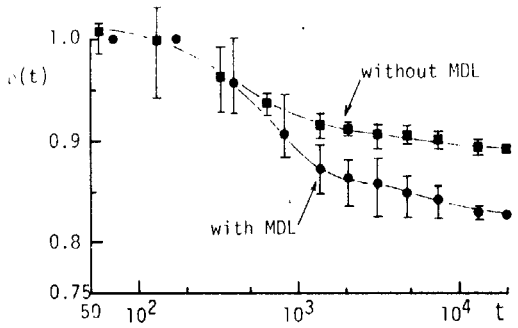
$$p(z) = \frac{c(z)}{c(\lambda)}$$

i.e.,

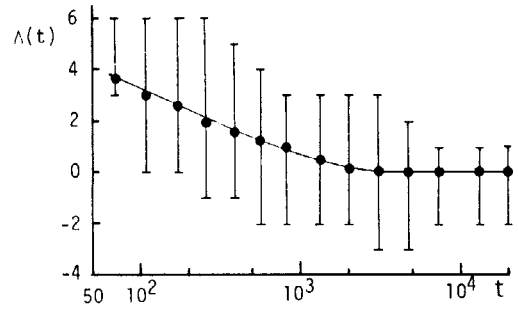
$$p(z) = \frac{c(0|z) + c(1|z)}{c(0|\lambda) + c(1|\lambda)} \tag{18}$$

The context $z = x(t - d + 1, t - 1)$ is moved from the root of the tree to the leaf or the node immediately before the virtual node; d minimizing Eq. (14) is determined, using Eqs. (17) and (18), in the range, and the encoding of x_t is performed by Eq. (15).

In example 2, on the other hand, it is not necessarily true that the count is made



(a) Entropy of Model 2



(b) Estimation errors of local order

Fig. 5. Results of model 2 for several sequences with H = 0.8.

for all symbols in $x(0, t-1)$ for the context λ. Consequently, the validity of the estimation by Eq. (16) is not ensured in general. As was discussed earlier, one can expect that the estimation (17) asymptotically approaches the true value, when the context z is either the true context or is deeper on the tree. However, in this paper, the estimation by Eqs. (17) and (18) is used directly as an approximation. Since the second term in the right-hand side of Eq. (14) corresponds to the error in the parameter estimation, a correction is needed according to the number of samples in the parameter estimation. Letting

$$j \triangleq c(\lambda) - 1 \tag{19}$$

t in example 3 is almost the same as j immediately before the encoding of x_t, and one can write Eq. (14) as

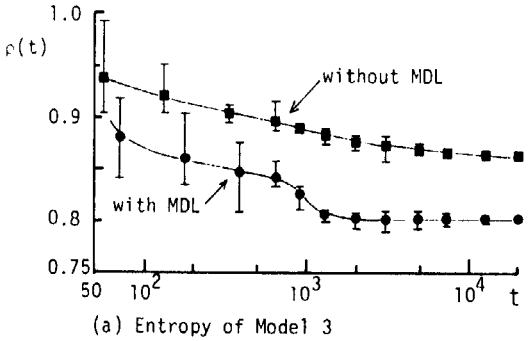
$$\rho_d(t) = H_d(t) + \frac{d}{2}\frac{\log j}{j} \tag{20}$$

Based on this property, Eq. (20) is introduced into example 2, as a function to be minimized corresponding to Eq. (14); j is the same as the number of substrings generated by the incremental parsing algorithm. The foregoing approximation and the correction are adopted as a result of the experiment in Sect. 4, as the means to arrive at a satisfactory result.

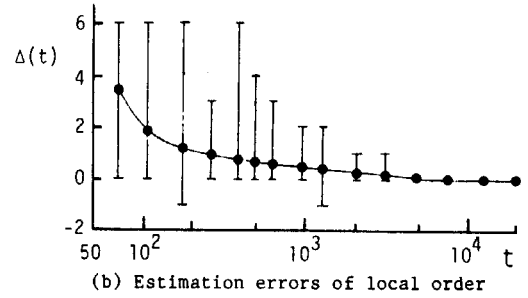
Table 1. Output source of experimental data

Context z	Local order	P(0|z)
00	3	0.2
10	3	0.8
11	3	0.7
101	4	0.3
0001	5	0.5
01001	6	0.2
11001	6	0.6

Entropy H = 0.80



(a) Entropy of Model 3



(b) Estimation errors of local order

Fig. 6. Results of model 3 for several sequences with H = 0.8.

4. Experiment and Discussions

4.1 Experiment by computer

To examine the effect of MDL criterion in examples 2 and 3, an experiment was performed with the input data generated by the assumption in this paper. In the following, for simplicity, the entropy of the model

$\rho(x(1, t))$ is written simply as $\rho(t)$. The difference between the true local order d_t^* for x_t and the order d_t estimated by the encoder is written as

$$\Delta(t) = d_t^* - d_t$$

As examples, Figs. 5 and 6 show $\rho(t)$ and $\Delta(t)$ for examples 2 and 3, respectively, for several sequences from the information source with entropy H = 0.8. In the model without using the MDL criterion, the local order is not estimated, and, consequently, $\Delta(t)$ is not shown.

It is seen in both examples 2 and 3 that, when the data length is larger than a certain value, the compression ratio is improved greatly by introducing the MDL criterion. Of course, the time until one arrives at the correct estimation of the local order depends on the data. The model with MDL criterion in example 3 estimates the order exactly. The model with MDL criterion in example 2 does not estimate the order very correctly, since the algorithm itself is based on the approximation. Still, the estimation seems to be satisfactory. It is not obvious in these figures but, in general, it is observed that the estimated local order does not change rapidly until the correct estimation is arrived at, rather than gradually increasing to the true value.

4.2 Discussion on sampler improvement

This paper fixed the sampler, and the encoder was designed in the optimum way. By contrast, the sampler may be reconstructed for the obtained encoder. One point to improve is that transformation of the tree structure is needed as was discussed in Sect. 3.3, since the tree structure of the sampler does not satisfy the requirement from the encoder. The second point is that the sampler may continue an excessive learning even if the encoder estimated the correct order, since there is no method to determine the situation.

In examples 2 and 3, the sampler performs the learning directly to maintain the relation of Eq. (1). While, the algorithm "context" by Rissanen [11], aiming at the relation of Eq. (2), is an improvement of the first point as a result, eliminating the transformation of the tree structure. The gross number of samples in Rissanen's method is $O(n \log n)$, which is the same as in example 3 in this paper. In fact, the compressions in both cases are almost the same for the encoder with the MDL criterion. Since, however, Rissanen's model does not explicitly introduce the MDL criterion, it cannot be considered as highly refined from the viewpoint of the computational technique.

Rissanen discussed further the second point. A simple convenient procedure is proposed, where the input sequence is partitioned into several blocks, and the upper limit of the estimation depth is set for each block, based on the result of estimation for the block up to that stage. However, a relation must exist between the structure of the input sequence and the sequence length until one arrives at the correct estimation of the local order. It will be necessary to discuss this point to explore a more refined method.

It is true that the sampler has room for improvement, but the sampler in example 3 is a very powerful one in the following sense. There exists the encoder of the original model with a high universality as well as the encoder with the MDL criterion which can realize a high compression performance for a certain class of input sequences. By contrast, for the algorithm context by Rissanen, the existence of the encoder with a high universality corresponding to the former has not been known, but there is a remakrable feature that the sorting function of the sequence can be introduced. The sorting function is not discussed further since it is not considered when the usual Markov property is assumed for the information source, as in this paper.

5. Conclusions

This paper proposed a model for the adaptive data compression, where the sampler and the encoder are separated. The learning structure and the encoding process of the existing universal codes are described. Furthermore, the introduction of the MDL criterion into the encoder is described, aiming at the improvement of the compression ratio, thereby fixing the sampler.

As a result, a remarkable improvement of the perfórmance was verified by experiment for the sequences generated by a Markov source, as is assumed in this paper. The sampler for the universal code proposed by the author has a powerful property in that both the encoder with a high universality and the encoder with the MDL criterion exist. However, this implies that a problem is produced as to what combination should be applied to the actual individual data. This is a problem left for further study, together with the discussion of the relation to the practical design techniques [21].

Acknowledgement. The author appreciates the suggestions of Prof. M. Iri, Univ. Tokyo and Prof. T.S. Han, Senshu Univ.

REFERENCES

1. D.A. Huffman. A method for the construction of minimum redundancy codes, Proc. IRE, 40, 9, pp. 1098-1101 (Sept. 1982).
2. M. Pechura. File archival techniques using data compression, Commun. ACM, 25, 9, pp. 605-609 (Sept. 1982).
3. R.G. Gallager. Variations on a theme by Huffman, I.E.E.E. Trans. Inf. Theory, IT-24, 6, pp. 668-674 (Nov. 1978).
4. J.S. Vitter. Design and analysis of dynamic Huffman coding, Proc. 26th I.E.E.E. Symp. on Foundations of Computer Science, pp. 293-302 (Oct. 1985).
5. J.Ziv and A. Lempel. Compression of individual sequences via variable-rate coding, I.E.E.E. Trans. Inf. Theory, IT-24, 5, pp. 530-536 (Sept. 1978).
6. H. Yamamoto and K. Nakada. On the improvement of Ziv-Lempel code and its evaluation by simulation, [II], Tech. Rep. I.E.C.E., Japan, CS84-135 (Jan. 1985).
7. H. Yokoo. An improved Ziv-Lempel coding scheme for universal source coding, Trans. (A), I.E.C.E., Japan, J68-A, 7, pp. 664-671 (July 1985).
8. Y. Honda and K. Aihara. A consideration on an incremental parsing for noiseless universal coding, Proc. 8th Meeting, Inf. Theory. & Appl., pp. 251-256 (Dec. 1985).
9. J. Rissanen. A universal prior for integers and estimation by minimum description length, Ann. Statist., 11, 2, pp. 416-431 (June 1983).
10. J. Rissanen. Universal coding, information, prediction, and estimation, I.E.E.E. Trans. Inf. Theory, IT-30, 4, pp. 629-636 (July 1984).
11. J. Rissanen. A universal data compression system, I.E.E.E. Trans. Inf. Theory, IT-29, 5, pp. 656-664 (Sept. 1983).
12. H. Yokoo. A new scheme of the Ziv-Lempel-type universal codes for sequential data compression, 8th Proc. 8th Meeting, Inf. Theor. & Appl., pp. 228-233 (Dec. 1985).
13. J. Rissanen. Generalized Kraft inequality and arithmetic coding, IBM J. Res. & Dev., 20, pp. 198-203 (May 1976).
14. M. Guazzo. A general minimum-redundancy source-coding algorithm, I.E.E.E. Trans. Inf. Theory, IT-26, 1, pp. 15-25 (Jan. 1980).
15. C.B. Jones. An efficient coding system for long source sequences, I.E.E.E. Trans. Inf. Theory, IT-27, 3, pp. 280-291 (May 1981).
16. J. Rissanen and G.G. Langdon,Jr. Universal modeling and coding, I.E.E.E. Trans. Inf. Theory, IT-27, 1, pp. 12-23 (Jan. 1981).
17. J.P.M. Schalkwijk. An algorithm for source coding, I.E.E.E. Trans. Inf. Theory, IT-18, 3, pp. 395-399 (May 1972).
18. J. Rissanen. Arithmetic codings as number representations, Acta Polytech. Scandinavica, Math., 31, pp. 44-51 (Dec. 1979).
19. J.G. Cleary and I.H. Witten. A comparison of enumerative and adaptive codes, I.E.E.E. Trans. Inf. Theory, IT-30, 2, pp. 306-315 (Mar. 1984).
20. G.G. Langdon, Jr. A note on the Ziv-Lempel model for compressing individual sequences, I.E.E.E. Trans. Inf. Theory, IT-29, 2, pp. 284-287 (Mar. 1983).
21. J.G. Cleary and I.H. Witten. Data compression using adaptive coding and partial string matching, I.E.E.E. Trans. Commun., COM-32, 4, pp. 396-402 (April 1984).

AUTHOR

Hidetoshi Yokoo graduated 1978 Dept. Math. Eng. and Instr. Phys., Fac. Eng., Univ. Tokyo. Completed Master's program 1980 Inf. Eng., Grad. School. Assistant, Dept. Electrical Eng., Fac. Eng., Yamagata Univ. Engaged in researches in data compression, symbol processing and their applications. Member, Inf. Proc. Soc. Japan.