Arithmetic Coding for Finite-State Noiseless Channels

Serap A. Savari[1] *Robert G. Gallager*[2]

Abstract: We analyze the expected delay for infinite precision arithmetic codes and suggest a practical implementation that closely approximates the idealized infinite precision model.

Index Terms: Arithmetic coding, expected delay analysis, finite precision arithmetic.

INTRODUCTION

Arithmetic coding is a powerful and conceptually simple data compression technique. The general idea in arithmetic coding is to map a sequence of source symbols into a point on the unit interval and then to represent this point as a sequence of code letters. P. Elias first conceived of this idea for the case of equal cost code letters and his technique could be considered a generalization of the Shannon-Fano code (see Shannon (1948)). Elias' encoding technique is ideal in the sense that it encodes exactly at the entropy rate; it is described briefly in Abramson (1963). Jelinek (1968) gave a more detailed exposition of Elias' code, explained how to implement a version of it which maintains finite buffers of source symbols and code letters, and demonstrated that arithmetic calculations must be accomplished with infinite accuracy in order to encode and decode arbitrarily long strings of source symbols. Thus, Elias' code in this ideal form is not a practical coding technique for very long strings of source symbols.

Rissanen (1976) found the first arithmetic code which does not suffer from the precision problem. In contrast to Elias' code, Rissanen's code involved a mapping of growing strings of source symbols into increasing non-negative integers; as a consequence, source symbols are decoded in a last-in, first-out manner, which is undesirable from the viewpoint of decoding delay. Pasco (1976) used similar ideas to create a practical arithmetic code for which source symbols are decoded first-in, first-out; his code is more reminiscent of Elias' arithmetic code. Rissanen and Langdon (1979) described other arithmetic codes and derived a duality result between first-in, first-out codes and last-in, first-out codes. The modifications that were introduced in these papers to account for the precision problem and make arithmetic coding a more practical

[1]Supported by an AT&T Bell Laboratories GRPW Fellowship and a Vinton Hayes Fellowship.
[2]Supported by NSF grant 8802991-NCR.

encoding scheme are complex and elusive to explain in an easy way; we refer the reader to Langdon (1984) for some perspective on these modifications. Jones (1984) and Witten et. al. (1987) discovered other practical arithmetic codes which are similar to the codes of Elias and Pasco.

Guazzo (1980) realized that arithmetic coding could be used to map source sequences into more general code alphabets than those with N equal cost letters. He described a practical arithmetic code which efficiently maps sequences of source symbols into sequences of letters from a channel with memoryless letter costs; i.e., the cost of transmitting any code letter depends only on that letter and different letters may have different transmission costs. Todd et. al. (1983) specified a practical arithmetic code to efficiently encode source sequences into sequences from a channel with finite-state letter costs; here, the cost of transmitting a code letter depends on the letter, the string of previously transmitted letters, and the state of the channel before transmission began.

In this paper, we provide an alternate approach to arithmetic coding by concentrating on the issue of coding delay. We will generalize Elias' code first to memoryless cost channels and later to finite-state channels and demonstrate that the expected value of coding delay is bounded for both types of channels. We also suggest a practical implementation that focuses on delay and is closely related to Elias' ideal arithmetic code. For the case of binary equal cost code letters, the expected delay analysis and an alternate implementation appeared earlier in course notes prepared by the second author.

MEMORYLESS COST CHANNELS

Assume a source emitting independent identically distributed symbols from a finite set $\{0, 1, \ldots, K-1\}$. The letter probabilities $p_0, p_1, \ldots, p_{K-1}$ are strictly positive. We initially assume a noiseless channel with memoryless letter costs; i.e., our channel is a device that accepts input from a specified set of letters, say $\{0, 1, \ldots, N-1\}$ with (positive) letter costs $c_0, c_1, \ldots, c_{N-1}$, respectively. The simplest and most common case is that of binary equal cost channel letters. The added generality here will permit an easy generalization to finite-state channels; these channels include the set of constrained channels such as those in magnetic disk storage devices. We shall also see later that we can easily dispense with the assumption that the source is memoryless.

Shannon (1948) demonstrated that for memoryless cost channels, the infimum, over all source coding techniques, of the expected cost per source symbol is equal to $\frac{H}{C}$, where $H = -\sum_{i=0}^{K-1} p_i \log_2 p_i$ is the *entropy* of the source and C, the *capacity* of the channel, is the real root of the equation $\sum_{i=0}^{N-1} 2^{-Cc_i} = 1$. However, he did not specify a technique to construct such codes.

We denote the random sequence produced by the source as $y = \{y_1, y_2, y_3, \ldots\}$ and let $y^{(m)} = \{y_1, y_2, \ldots, y_m\}$ for $m \geq 0$. Since the source is memoryless, $P[y^{(m)}] = \prod_{j=1}^{m} P[y_j]$ where $P[y_j = k] = p_k,\ 0 \leq k \leq K-1$.

The idea in arithmetic coding is to map $y^{(m)}$ into a subinterval of the unit interval in such a way that as m increases, the corresponding subinterval shrinks to a point $x(y)$. The resulting subintervals are then represented by channel strings $z^{(n)} = \{z_1, z_2, \ldots, z_n\}$ that grow into the output channel sequence $z = \{z_1, z_2, \ldots\}$. First, we discuss the mapping of source strings into subintervals of the unit interval. Let $\mathcal{I}(y^{(m)})$ denote the subinterval corresponding to source string $y^{(m)}$; $y^{(0)}$ denotes the null source string. As in earlier work on arithmetic coding, the mapping of source strings into (left half-closed) intervals has been selected to satisfy two requirements. The first is that for all source strings $y^{(m)}$, the width of interval $\mathcal{I}(y^{(m)})$ is equal to the a priori probability of $y^{(m)}$. The other property is that for any source string $y^{(m)}$, we have that $\mathcal{I}(y^{(m)}, 0)\ , \ldots , \mathcal{I}(y^{(m)}, K-1)$ are disjoint intervals whose union is $\mathcal{I}(y^{(m)})$. For the null string, $\mathcal{I}(\emptyset)$ is the unit interval.

One way to implement these requirements is as follows. We define

$$f(i) = f_1(i) = \sum_{j=0}^{i-1} p_j \tag{1}$$

$$f(y^{(m)}) = f(y^{(m-1)}) + f_1(y_m) \cdot P(y^{(m-1)}),\ m > 1 \tag{2}$$

$$\text{and } \mathcal{I}(y^{(m)}) = [f(y^{(m)}), f(y^{(m)}) + P(y^{(m)})). \tag{3}$$

Figure 1 illustrates this procedure. We note that the mapping of source sequences to points has the following monotonicity property: Given arbitrary distinct source sequences u and v, $x(u) > x(v)$ if and only if u is lexicographically larger than v.

Next consider the mapping of points on the unit interval into strings of channel letters. We note that the mapping of source sequences to points defined by (1), (2) and (3) is, for all

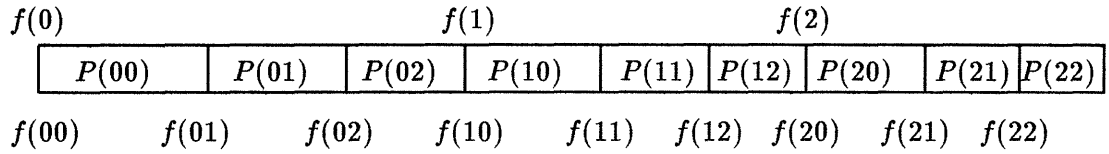

Figure 1:

practical purposes, invertible. We shall use a related technique to map strings of channel letters into subintervals of the unit interval; we will see later how to combine the mapping of source strings into intervals with the mapping of intervals into channel strings. Let $z^{(n)}$ denote the initial string $z^{(n)} = \{z_1, \ldots, z_n\}$ and $\mathcal{J}(z^{(n)})$ denote the subinterval corresponding to this string; as before, $z^{(0)}$ represents the null channel string. Guazzo (1980) established a duality between the mapping of source strings into intervals and the mapping of channel strings into intervals; more specifically, he showed that it is optimal to associate a probability 2^{-Cc_i} with each channel letter i and then to map channel strings into subintervals in exactly the same way that source strings are mapped into subintervals. Therefore, if for any channel string $z^{(n)}$, $c(z^{(n)})$ and $l(z^{(n)})$ denote the cost of transmitting $z^{(n)}$ and the length of $\mathcal{J}(z^{(n)})$, respectively, then we require that for all channel strings $z^{(n)}$, $l(z^{(n)}) = 2^{-C \cdot c(z^{(n)})}$ and $\mathcal{J}(z^{(n)}, 0)$, ..., $\mathcal{J}(z^{(n)}, N-1)$ are disjoint intervals whose union is $\mathcal{J}(z^{(n)})$. The convention for the null symbol is that $\mathcal{J}(\emptyset) = [0, 1)$.

To satisfy these requirements, we employ a mapping that is analogous to the mapping we used for source strings. We define

$$g(i) = g_1(i) = \sum_{j=0}^{i-1} l(j) = \sum_{j=0}^{i-1} 2^{-Cc_j} \tag{4}$$

$$g(z^{(n)}) = g(z^{(n-1)}) + g_1(z_n) \cdot l(z^{(n-1)}) = g(z^{(n-1)}) + g_1(z_n) \cdot 2^{-C \cdot c(z^{(n-1)})}, \quad n > 1 \tag{5}$$

$$\text{and } \mathcal{J}(z^{(n)}) = [g(z^{(n)}),\ g(z^{(n)}) + l(z^{(n)})) = [g(z^{(n)}),\ g(z^{(n)}) + 2^{-C \cdot c(z^{(n)})}). \tag{6}$$

Clearly, the mapping of channel sequences to points has the same lexicographic property as the mapping of source sequences to points.

For the inverse mapping, if we are given any subinterval $\mathcal{X}$ of the unit interval, the channel string associated with $\mathcal{X}$ is the longest string $z^{(n)}$ for which $\mathcal{J}(z^{(n)})$ contains $\mathcal{X}$.

We now have the tools to discuss the encoding of source sequence y. On observing $y^{(m)}$,

the encoder knows that the limit point $x(y)$ lies in the interval $\mathcal{I}(y^{(m)})$. Thus, if $\mathcal{I}(y^{(m)})$ is contained in $\mathcal{J}(z^{(n)})$ for some channel string $z^{(n)}$, then the encoder can emit $z^{(n)}$ as the first n letters of z. Hence, as the source emits successive letters y_m, the intervals $\mathcal{I}(y^{(m)})$ shrink and more channel letters can be emitted.

To demonstrate the efficiency of the above procedure, we would like to show that when the source has emitted $y^{(m)}$, the encoder will have issued a channel string $z^{(n)}$ with cost of transmission close to $\frac{I(y^{(m)})}{C}$, where $I(y^{(m)}) = -\log_2(P[y^{(m)}])$, and that $z^{(n)}$ will be sufficient for the decoder to decode all but the last few letters of $y^{(m)}$. We first consider the number of letters $m(n)$ that the source must emit in order for the encoder to issue the first n channel letters. Since $P(y^{(m(n))})$ is the length of $\mathcal{I}(y^{(m(n))})$, $2^{-C\cdot[cost\ of\ z^{(n)}]}$ is the length of $\mathcal{J}(z^{(n)})$, and $\mathcal{I}(y^{(m(n))})$ is contained in $\mathcal{J}(z^{(n)})$,

$$P(y^{(m(n))}) \leq 2^{-C\cdot[cost\ of\ z^{(n)}]}. \tag{7}$$

Taking the logarithm of both sides of (7) and dividing the resulting inequality by $-C$ gives

$$cost\ of\ z^{(n)} \leq \frac{1}{C} I(y^{(m(n))}). \tag{8}$$

Since this inequality can be arbitrarily loose, we want to show that for each n,
$E\left(\frac{I(y^{(m(n))})}{C} - [cost\ of\ z^{(n)}]\right)$ is bounded.

In order to accomplish this, let $z^{(n)}$ be fixed and let x be the final encoded point. The point x, conditional on $z^{(n)}$, is a uniformly distributed random variable in the interval $\mathcal{J}(z^{(n)})$, but we initially regard it as a fixed value. Define $D(x)$ as the distance between x and the nearest endpoint of $\mathcal{J}(z^{(n)})$ (see Figure 2). We note that the point x must be contained in $\mathcal{I}(y^{(m)})$ for all m. Also, since $m(n)$, by definition, is the smallest m for which $\mathcal{J}(z^{(n)})$ contains $\mathcal{I}(y^{(m)})$, we see that $\mathcal{I}(y^{(m(n)-1)})$ must contain one of the endpoints of $\mathcal{J}(z^{(n)})$ as well as x and thus must have width of at least $D(x)$. Hence, for the given $z^{(n)}$ and x, $P(y^{(m(n)-1)}) \geq D(x)$ and therefore

$$I(y^{(m(n)-1)}) \leq -\log_2(D(x)). \tag{9}$$

Now consider x as a random variable uniformly distributed over $\mathcal{J}(z^{(n)})$. $D(x)$ is then uniformly

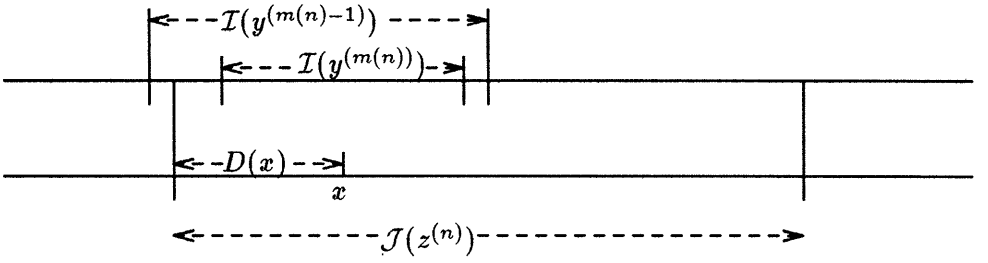

Figure 2:

distributed between zero and half the length of $\mathcal{J}(z^{(n)})$. Using (9), we see that

$$E[I(y^{(m(n)-1)} \mid z^{(n)})] \leq -E[\log_2(D(x)) \mid z^{(n)}]. \tag{10}$$

Since $D(x)$ is uniformly distributed, we have that

$$\begin{aligned} E[\log_2(D(x)) \mid z^{(n)}] &= \int_{D=0}^{\frac{1}{2}\cdot 2^{-C\cdot c(z^{(n)})}} 2 \cdot 2^{C\cdot[cost\ of\ z^{(n)}]}(\log_2 D)\, dD \\ &= -C \cdot [cost\ of\ z^{(n)}] - \log_2(2e). \end{aligned} \tag{11}$$

Hence,

$$cost\ of\ z^{(n)} \geq \frac{1}{C} E[I(y^{(m(n)-1)} \mid z^{(n)})] - \frac{1}{C}\log_2(2e). \tag{12}$$

If p_{min} is the probability of the least likely source symbol, then for all $y^{(m)}$,

$$I(y^{(m)}) = I(y^{(m-1)}) + I(y_m) \leq I(y^{(m-1)}) + \log_2(\frac{1}{p_{min}}). \tag{13}$$

Therefore, (12) and (13) imply that

$$cost\ of\ z^{(n)} \geq \frac{1}{C} E[I(y^{(m(n))} \mid z^{(n)})] - \frac{1}{C}\log_2\left(\frac{2e}{p_{min}}\right). \tag{14}$$

We note that the above inequality is uniformly true for all $z^{(n)}$ and all n. (14) and (8) imply that the encoder generates cost, on the average, with the ideal of $\frac{H}{C}$ per source symbol; however, there is a slight deficit in the cost of each code string that is produced since the encoder is storing the most recent information about the source sequence in order to correctly emit the

next few channel letters. This deficiency in cost becomes increasingly insignificant as we average over longer and longer source strings.

We can use a very similar argument to bound the delay between the generation of channel letters at the decoder and the generation of decoded source symbols. For an arbitrary source string $y^{(m)}$, we let $n(m)$ denote the number of code letters that must be received at the decoder in order for the string $y^{(m)}$ to be decoded. Using a very similar derivation to that above and letting c_{max} denote the maximum channel letter cost, it is straightforward to show that

$$E[cost\ of\ z^{(n(m))}\ \mid y^{(m)}] \leq \frac{1}{C} I(y^{(m)}) + \frac{1}{C} \log_2 \left(\frac{2e}{2^{-C c_{max}}} \right). \tag{15}$$

We now combine (14) and (15). Consider a given string $y^{(m)}$ out of the decoder, and suppose that $z^{(n(m))}$ is the required code string to decode $y^{(m)}$. Figure 3 demonstrates the relationship between $\mathcal{I}(y^{(m)})$, $\mathcal{J}(z^{(n(m))})$ and $\mathcal{I}(y^{(m')})$ where $y^{(m')}$ is the extended source string required to produce $z^{(n(m))}$. Conditional on both $y^{(m)}$ and $z^{(n(m))}$, we see that x is uniformly distributed

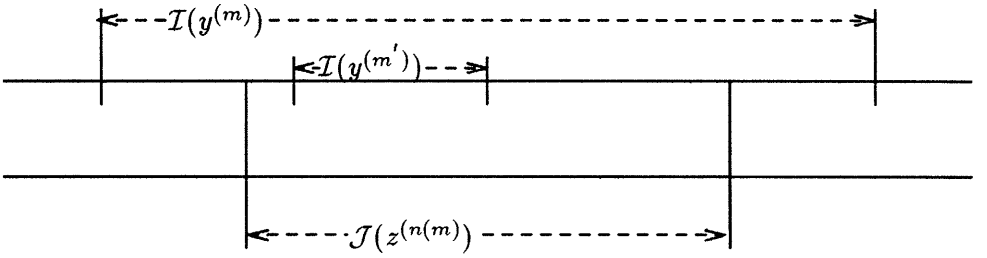

Figure 3:

over $\mathcal{J}(z^{(n(m))})$, and thus $y^{(m')}$ satisfies (from (14))

$$cost\ of\ z^{(n(m))}\ \geq \frac{1}{C} E[I(y^{(m')} \mid z^{(n(m))})] - \frac{1}{C} \log_2 \left(\frac{2e}{p_{min}} \right). \tag{16}$$

Using (15) to take the expected value of this over $z^{(n(m))}$, we see that for any given $y^{(m)}$, the expected self-information of the extended source string $y^{(m')}$ required from the source to produce the $n(m)$ channel letters needed to decode $y^{(m)}$ satisfies

$$E[I(y^{(m')} \mid y^{(m)})] - I(y^{(m)}) \leq \log_2 \left(\frac{4e^2}{p_{min} \cdot 2^{-C c_{max}}} \right). \tag{17}$$

The expectation here is over the source symbols y_{m+1}, y_{m+2}, ... for the given string $y^{(m)}$. It is important to note that the bound does not depend on m or $y^{(m)}$. The upper bound in (17) states that on average there is very little delay from the encoder to the decoder. To convert this bound into a bound on the number of letters $m' - m$, let p_{max} be the maximum source letter probability. Then $\log_2\left(\frac{1}{p_{max}}\right)$ is the minimum possible self-information per source letter and

$$E[m' - m \mid y^{(m)}] \leq \frac{\log_2\left(\frac{4e^2}{p_{min}\cdot 2^{-Cc_{max}}}\right)}{\log_2\left(\frac{1}{p_{max}}\right)}. \tag{18}$$

In Appendix I, we generalize the above analysis to obtain an upper bound on the moment generating function for the delay distribution and subsequently show that there exists a constant $\mathcal{K}$ for which

$$P(m' - m \geq k \mid y^{(m)}) \leq \mathcal{K} \cdot k^2 \cdot p_{max}^k. \tag{19}$$

Implementation

In actual implementation, it is not possible to calculate the intervals used in encoding and decoding exactly. We view the arithmetic as being performed using binary fixed point arithmetic with M binary digits of accuracy. Assume that $2^{-M} << \min\{p_{min}, 2^{-Cc_{max}}\}$. There is some flexibility in how numbers are rounded to M bits, but it is vital that the encoder and decoder use exactly the same rule and the rounding be done at the appropriate time. In order to mitigate the effects of round-off, we will use a two-part arithmetic coder which is outlined in Figure 4. The outer arithmetic coder will map source sequences into binary sequences with memoryless,

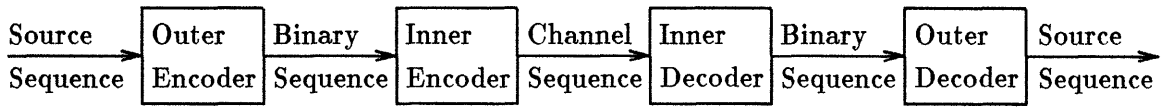

Figure 4:

unit digit costs. We let x_b represent the point on the unit interval corresponding to the source sequence y and $b = \{b_1, b_2, \ldots\}$ be the corresponding binary sequence. The capacity of this binary channel is easily seen to be equal to one; therefore, our earlier results show that over the long term, the average number of binary digits per source symbol (for infinite precision

arithmetic coding) is H. Furthermore, since the mapping from source sequences to points on the unit interval is done so that the random variable b is uniformly distributed on the real line, each of the digits $b_1, b_2, \ldots$ in the binary expansion of b is independent and equiprobably equal to 0 or 1. The inner arithmetic coder will map the binary sequence b into a sequence of letters from the original channel alphabet. Since $b_1, b_2, \ldots$ are independent and equiprobably equal to 0 or 1, the entropy of the incoming binary sequence is 1. As before, the capacity of the channel is C. Hence, our earlier conclusions indicate that the second encoder generates cost, on the average, with the ideal of $\frac{1}{C}$ per binary digit. Combining these averages, we see that over a large source string, this double encoding procedure generates cost, on the average, with the ideal of $\frac{H}{C}$ per source symbol. Therefore, in theory, we do not lose efficiency by splitting the coder into these two parts. However, it seems likely that there will be an increase in expected delay because coding is done in two steps. By using (17) twice, i.e., for the outer and inner coders, we see that the new upper bound on the delay between encoding and decoding is

$$E[m' - m \mid y^{(m)}] \leq \frac{\log_2\left(\frac{4e^2}{p_{min}\cdot\frac{1}{2}}\right) + \log_2\left(\frac{4e^2}{\frac{1}{2}\cdot 2^{-Cc_{max}}}\right)}{\log_2\left(\frac{1}{p_{max}}\right)} = \frac{\log_2\left(\frac{64e^4}{p_{min}\cdot 2^{-Cc_{max}}}\right)}{\log_2\left(\frac{1}{p_{max}}\right)}. \qquad (20)$$

We first discuss the behavior of the outer arithmetic coder. For the sake of simplicity, we begin with an algorithm that is not entirely correct. The outer encoder receives one source symbol at a time and calculates the corresponding interval with accuracy to M bits. Since $P(y^{(m)})$ is approaching 0 with increasing m, it is essential that the intervals be renormalized as binary digits are emitted. Every time a source symbol is read in, the encoder issues the longest binary string whose matching interval contains the current normalized source string interval. If the length of this binary string is l, the encoder renormalizes by expanding the fraction of the unit interval which contains the source string interval by a factor of 2^l. This causes $p_{norm}(y^{(m)})$ to be multiplied by 2^l and $f_{norm}(y^{(m)})$ to be set to the fractional part of 2^l times the original value of $f_{norm}(y^{(m)})$.

More precisely, the outer encoder keeps in its memory a normalized interval starting at $f_{norm}(y^{(m)})$ and of width $p_{norm}(y^{(m)})$. We denote the right endpoint of this interval by $e_{norm}(y^{(m)})$. Initially, $f_{norm}(\emptyset) = 0$, $p_{norm}(\emptyset) = 1$ and $m = 1$. In order to ensure that the intervals corresponding to different m tuples $y^{(m)}$ are disjoint and have $[0, 1)$ as their union, the interval end

points are calculated directly and $p_{norm}(y^{(m)})$ is taken as the difference between the end points. The outer encoder employs the following algorithm.

1. Accept y_m into the encoder.

2. Calculate the new interval as follows:

$$f_{norm}(y^{(m)}) = f_{norm}(y^{(m-1)}) + f_1(y_m) \cdot p_{norm}(y^{(m-1)}) \quad (21)$$

$$e_{norm}(y^{(m)}) = f_{norm}(y^{(m-1)}) + f_1(y_m + 1) \cdot p_{norm}(y^{(m-1)}) \quad (22)$$

$$p_{norm}(y^{(m)}) = e_{norm}(y^{(m)}) - f_{norm}(y^{(m)}) \quad (23)$$

$$\mathcal{I}_{norm}(y^{(m)}) = [f_{norm}(y^{(m)}),\ f_{norm}(y^{(m)}) + p_{norm}(y^{(m)})) \quad (24)$$

   To use (22) when $y_m = K - 1$, we use the convention that $f_1(K) = 1$. Equations (21) and (22) will be replaced later by (25) to (28).

3. Find the longest binary string $B^{(l)} = \{B_1, \ldots, B_l\}$ for which $\mathcal{I}_{norm}(y^{(m)}) \subset [\sum_{i=1}^{l} B_i 2^{-i},\ \sum_{i=1}^{l} B_i 2^{-i} + 2^{-l})$. Possibly, $B^{(l)} = \emptyset$, $l = 0$.

4. Emit the binary string $B^{(l)}$ as output.

5. Renormalize by

$$f_{norm}(y^{(m)}) = 2^l f_{norm}(y^{(m)}) - \lfloor 2^l f_{norm}(y^{(m)}) \rfloor$$

$$p_{norm}(y^{(m)}) = 2^l p_{norm}(y^{(m)})$$

6. Increment m and goto step 1.

The purpose of step 5 is to eliminate the more significant binary digits that are no longer needed in the encoding and decoding and to add less significant digits that increase the precision as the intervals shrink. Note that renormalization is achieved with no additional round-off errors.

In order to see why this encoding algorithm does not operate correctly, we consider the example of a ternary equiprobable source. First we examine the behavior of the encoder when the input consists of a long string of repetitions of the symbol 1. Without round-off errors,

$\mathcal{I}_{norm}(y^{(1)}) = [\frac{1}{3}, \frac{2}{3})$, $\mathcal{I}_{norm}(y^{(2)}) = [\frac{4}{9}, \frac{5}{9})$, and in general, $\mathcal{I}_{norm}(y^{(m)}) = [\frac{1-3^{-m}}{2}, \frac{1+3^m}{2})$. Thus, for this string, $\mathcal{I}_{norm}(y^{(m)})$ continues to straddle the point $\frac{1}{2}$ and no binary digits are emitted by the encoder. Because arithmetic is performed with only M binary digits of accuracy, the left and right ends of these intervals must each be multiples of 2^{-M} and also must get close to $\frac{1}{2}$. For example, if the rounded off version of $\mathcal{I}_{norm}(y^{(m-1)})$ is $[\frac{1}{2} - 2^{-M}, \frac{1}{2} + 2^{-M})$, then no binary digit can be emitted, and since the length of $\mathcal{I}_{norm}(y^{(m-1)})$ is equal to $2 \cdot 2^{-M}$, it is impossible to split the interval into three distinct intervals to account for all possibilities of y_m. We will prevent this problem by changing the endpoints of certain intervals. The first revision is applicable for source intervals $\mathcal{I}_{norm}(y^{(m)})$ that straddle the point $\frac{1}{2}$ and have the property that the left endpoint is close to $\frac{1}{2}$. To facilitate renormalization, we move the left endpoint of this interval to $\frac{1}{2}$. This also allows a binary digit to be emitted. Let L be the largest integer for which $2^{-L} \geq \max\left(\frac{6 \cdot 2^{-M}}{p_{min}}, \frac{6 \cdot 2^{-M}}{2 - C_{max}}\right)$. We replace (21) with the following:

If $\frac{1}{2} - 2^{-L} \leq f_{norm}(y^{(m-1)}) + f_1(y_m) \cdot p_{norm}(y^{(m-1)}) < \frac{1}{2}$
and $f_{norm}(y^{(m-1)}) + f_1(y_m + 1) \cdot p_{norm}(y^{(m-1)}) > \frac{1}{2}$,

$$\text{then } f_{norm}(y^{(m)}) = \frac{1}{2} \tag{25}$$

$$\text{else } f_{norm}(y^{(m)}) = f_{norm}(y^{(m-1)}) + f_1(y_m) \cdot p_{norm}(y^{(m-1)}) \tag{26}$$

Since we are interested in producing a one-to-one onto mapping from the set of source sequences to the set of binary sequences, we must compensate for the truncation of any source string interval $\mathcal{I}_{norm}(z^{(n)})$. Here, we lengthen the interval of the string lexicographically preceding $z^{(n)}$ by relocating the right endpoint for that string's interval to $\frac{1}{2}$. We bring this about by changing (22) to:

If $\frac{1}{2} - 2^{-L} \leq f_{norm}(y^{(m-1)}) + f_1(y_m + 1) \cdot p_{norm}(y^{(m-1)}) < \frac{1}{2}$
and $f_{norm}(y^{(m-1)}) + f_1(y_m + 2) \cdot p_{norm}(y^{(m-1)}) > \frac{1}{2}$,

$$\text{then } e_{norm}(y^{(m)}) = \frac{1}{2} \tag{27}$$

$$\text{else } e_{norm}(y^{(m)}) = f_{norm}(y^{(m-1)}) + f_1(y_m + 1) \cdot p_{norm}(y^{(m-1)}) \tag{28}$$

Note that if the first condition above is satisfied, then $y_m + 2 \leq K$. Figure 5 illustrates the alterations. We selected L to ensure that the smallest interval that can straddle the point $\frac{1}{2}$ has

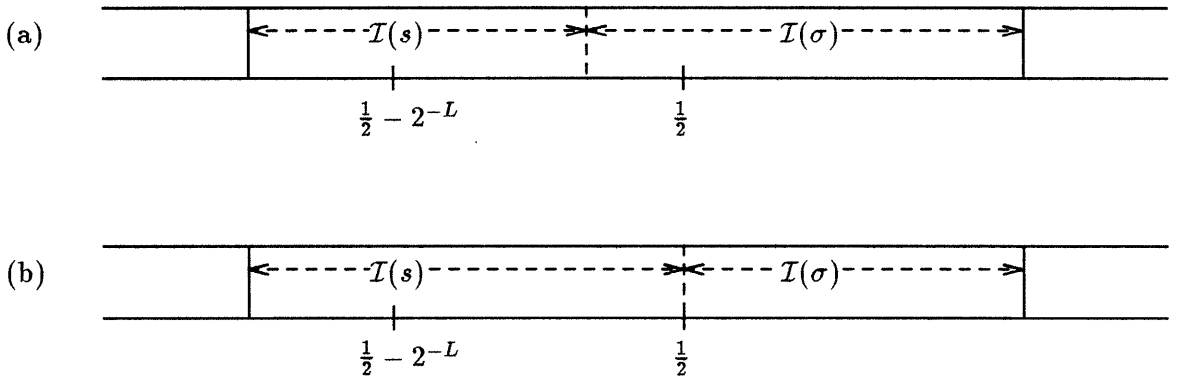

Figure 5:

Figure 5 illustrates the modification of adjoining intervals $\mathcal{I}(s)$ and $\mathcal{I}(\sigma)$ when $\mathcal{I}(\sigma)$ is an interval which straddles the point $\frac{1}{2}$ and has its left endpoint between $\frac{1}{2} - 2^{-L}$ and $\frac{1}{2}$. The left endpoint of $\mathcal{I}(s)$ is arbitrary.

length at least $\frac{6 \cdot 2^{-M}}{p_{min}}$ to guarantee that the next source symbol to enter the encoder will receive a non-zero interval size without any unusual round-off rules. When we discuss the inner coder, it will become clear why we also insist on having $2^{-L} \geq \frac{6 \cdot 2^{-M}}{2^{-C_{max}}}$. As a result of the modifications to (21) and (22), the binary output does not consist of digits that are equiprobably 0 or 1; however, for large M, it is fairly accurate to model the binary sequence in this way.

The above modifications are but one of many possible ways to handle the rarely occurring problem of normalized intervals that continue to straddle the point $\frac{1}{2}$. The only requirement in treating this issue is that the mapping from source sequences to binary sequences must be one-to-one onto.

We observe that when source string intervals are straddling the interval $[\frac{1}{2} - 2^{-L}, \frac{1}{2}]$, we experience some bounded delay in emitting binary digits and renormalizing. In all of the implementation schemes described in Langdon (1984), the binary output corresponding to $y^{(m)}$ is an approximation to the binary representation of $f(y^{(m)})$ and hence, binary digits are emitted more frequently than in the scheme we have described above. However, since the point $x(y)$ can appear anywhere in the interval $[f(y^{(m)}), f(y^{(m)}) + p(y^{(m)}))$, there is often a *carry-over*

problem resulting from the fact that several of the digits in the binary representation of $f(y^{(m)})$ may differ from the corresponding digits associated with $f(y^{(m+1)})$; in this scenario, it is either necessary to go back and correct the output or to insert bits in appropriate parts of the output to ensure that the carry-over problem does not affect the output ahead of the stuffed bits.

In Appendix II, we demonstrate that the outer encoder generates binary digits with a redundancy that decreases exponentially in M. This result holds for sources with memory also.

We next consider the outer decoder. The decoder decodes one source symbol at a time and maintains both a queue of incoming binary digits and a replica of the encoder. Initially, $m = 1$ and the queue is empty. The decoder, in attempting to decode y_m, uses the same rules as the encoder to calculate $f_{norm}(y^{(m)})$ and $p_{norm}(y^{(m)})$ for all choices of y_m given $y^{(m-1)}$. As new binary digits enter the queue, we can consider the queued letters as a normalized binary fraction of j significant bits, where j is the queue length. The decoder continues to read in binary digits one by one until the interval corresponding to this fraction lies within one of the K normalized intervals calculated above; at that point, the decoder decodes y_m, enters y_m into the replica encoder, deletes the binary digits which give no further information about the rest of the source sequence from the front of the queue (i.e., it will delete $\lfloor \log_2(\frac{1}{p_{norm}}) \rfloor$ bits), and renormalizes f_{norm} and p_{norm} by the encoder rules. It then increments m and repeats the above procedure.

We note that when y_m enters the encoder, the interval end points are calculated to M binary digits of accuracy. Therefore, after the encoder emits M binary digits, the resulting interval must have size 2^{-M} and thus y_m is decodable at this point, if not before. Hence, decoding always occurs with at most M binary digits in the queue. Therefore, by increasing M, we trade off smaller maximum delays between encoding and decoding for additional efficiency in terms of smaller round-off errors.

We now turn to the inner arithmetic coder which maps strings of binary digits into strings of channel letters. This coder functions independently of the outer coder. As we mentioned earlier, Guazzo associated a probability 2^{-Cc_i} with each channel letter i and then mapped channel strings into subintervals of the unit interval in exactly the same way that source strings are mapped into subintervals, except that the set of channel letter probabilities are used instead of the set of source letter probabilites. We can again capitalize on that idea here. We

saw that the outer coder created a one-to-one onto mapping of source sequences to binary sequences. For the inner coder, we need a one-to-one onto mapping between binary sequences and channel sequences. We can use the technique employed by the outer coder to produce a one-to-one onto mapping of channel sequences to binary sequences by using the set of probabilities $\{2^{-Cc_0}, \ldots, 2^{-Cc_{N-1}}\}$ instead of $\{p_0, \ldots, p_{K-1}\}$. Since the inner encoder maps arbitrary binary strings into strings of channel letters, its analogue in the outer coder is the outer decoder, which maps binary strings into strings of source symbols. The one-to-one onto nature of the encoding guarantees that the mapping of any binary string into a string of source symbols or channel letters is well-defined and that the inverse mapping will lead back to the original binary string. Similarly, by referring to Figure 4, we see that the counterpart of the inner decoder in the outer coder is the outer encoder. This duality between the inner and outer coders is the reason that we had selected L to satisfy $2^{-L} \geq \max\left(\frac{6 \cdot 2^{-M}}{p_{min}}, \frac{6 \cdot 2^{-M}}{2^{-Cc_{max}}}\right)$.

FINITE-STATE CHANNELS

We now generalize the preceding analysis and implementation to handle finite-state channels. A finite-state channel with finite alphabet $\{0, \ldots, N-1\}$ and set of states $\{0, \ldots, R-1\}$ is defined by specifying

1. for each pair (s, j), $0 \leq s \leq R-1$, $0 \leq j \leq N-1$, the cost $c_{s,j} \in [0, \infty]$ of transmitting j when the state is s

2. the state $S[s, j]$ after channel letter j is transmitted, given that the state of the channel is s prior to transmission.

The second rule inductively specifies the final state after an arbitrary channel string $z^{(n)}$ is transmitted from initial state s_0, and we denote this state by $S[s_0, z^{(n)}]$. As before, we assume a discrete memoryless source.

Let Z^* denote the set of all strings of channel letters. We say that $z^{(n)} \in Z^*$ is an element of Z_s^* if the cost of transmitting $z^{(n)}$ is finite given that the channel is in state s immediately before the first letter of $z^{(n)}$ is transmitted.

Let $c_s^* = \min_j c_{s,j}$. We allow the possibility of $c_s^* = 0$ for some state s, but assume that for every state s and every channel string $z^{(n)} \in Z_s^*$ with $n \geq R$, the cost of transmitting $z^{(n)}$ is

strictly positive.

We say that a finite-state channel is *irreducible* if for each pair of states s and s', there is a string $z^{(n)} \in Z_s^*$ for which $z^{(n)}$ drives the channel to state s' given that the channel was in state s prior to the transmission of the first letter of $z^{(n)}$; i.e., $S[s, z^{(n)}] = s'$. All finite-state channels that we will discuss are assumed to be irreducible.

We let $Z_{s'}(s) = \{\text{channel letters } j : S[s,j] = s'\}$ and for $\omega \geq 1$ we let $\mathcal{A}(\omega)$ denote the $R \times R$ matrix $\mathcal{A}(\omega) = [a_{s,s'}(\omega)]$ where $a_{s,s'}(\omega) = \sum_{j \in Z_{s'}(s)} \omega^{-c_{s,j}}$. To include $\omega = 1$, we use the convention that $1^{-\infty} = 0$. Shannon (1948) and Csiszár (1969) showed that there is a unique real number $\omega_0 \geq 1$ for which the greatest positive eigenvalue of $\mathcal{A}(\omega_0)$ equals one; furthermore, if $C = \log_2 \omega_0$, then the infimum, over all source coding techniques, of the expected cost per source symbol is equal to $\frac{H}{C}$. For this reason, we again refer to C as the capacity of the channel.

We assume that both the encoder and decoder know the initial state of the channel. For any channel string $z^{(n)}$ and any state s, let $c(s, z^{(n)})$, $S[s, z^{(n)}]$, $\mathcal{J}_s(z^{(n)})$ and $l(s, z^{(n)})$ denote the cost of transmitting $z^{(n)}$, the state of the channel after transmitting $z^{(n)}$, the subinterval corresponding to $z^{(n)}$, and the length of this subinterval, respectively, given the channel is in state s immediately before transmission begins.

Let $\mathcal{A} = [a_{s,s'}] = \mathcal{A}(\omega_0)$. Since $\mathcal{A}$ is a non-negative irreducible matrix with largest real eigenvalue equal to one, the Frobenius theorem implies that there exists a positive vector

$$\mathbf{v} = \begin{pmatrix} v_0 \\ \vdots \\ v_{R-1} \end{pmatrix}$$

for which

$$\mathbf{v} = \mathcal{A}\mathbf{v}. \tag{29}$$

In other words, for all $s \in \{0, \ldots, R-1\}$, we have

$$\sum_{s'=0}^{R-1} a_{s,s'} v_{s'} = \sum_{s'=0}^{R-1} \sum_{j \in Z_{s'}(s)} v_{s'} \omega_0^{-c_{s,j}} = v_s. \tag{30}$$

The normalization of $\mathbf{v}$ is not important since we will be using the ratios of components of $\mathbf{v}$.

We set up a mapping h from the Cartesian product of channel strings and channel states to subintervals of the unit interval as follows: if the channel is in state s_0 before transmission begins, then for any channel letter i and state s, we let

$$h_1(s,i) = \sum_{j=1}^{i-1} \frac{v_{S[s,j]}}{v_s}\, \omega_0^{-c_{s,j}} \tag{31}$$

$$h(i) = h_1(s_0,i) \tag{32}$$

For $m > 1$, given $z^{(m)}$, we define

$$h(z^{(m)}) = h(z^{(m-1)}) + h_1(S[s_0, z^{(m-1)}], z_m)\, \omega_0^{-c(s_0,z^{(m-1)})}. \tag{33}$$

Todd et. al. (1983) pointed out that it is appropriate and consistent to map each each channel string $z^{(m)}$ into the subinterval

$$\mathcal{J}_{s_0}(z^{(m)}) = [h(z^{(m)}),\ h(z^{(m)}) + \frac{v_{S[s_0,z^{(m)}]}}{v_{s_0}}\, \omega_0^{-c(s_0,z^{(m)})}\ ). \tag{34}$$

Note that the mapping formed by equations (31) to (34) reduces to the mapping defined by equations (4) to (6) in the special case of a memoryless cost channel. Since the length of $\mathcal{J}_{s_0}(z^{(m)})$ is $\frac{v_{S[s_0,z^{(m)}]}}{v_{s_0}}\, \omega_0{}^{-c(s_0,z^{(m)})} = \prod_{i=1}^{m} \frac{v_{S[s_{i-1},z_i]}}{v_{s_{i-1}}}\, \omega_0{}^{-c_{s_{i-1},z_i}}$, where s_i is defined inductively by $s_i = S[s_{i-1}, z_i]$, we see that when the channel is in state s, we can associate each channel letter j with a probability $\frac{v_{S[s,j]}}{v_s}\, \omega_0^{-c_{s,j}}$ and a next state $S[s,j]$.

The encoding of source sequence y follows the same procedure we used earlier given a memoryless cost channel; namely, if $\mathcal{I}(y^{(m)})$ is contained in $\mathcal{J}_{s_0}(z^{(n)})$ for some channel string $z^{(n)} \in Z^*_{s_0}$, then the encoder can emit $z^{(n)}$ as the first n letters of z. We observe that if $z^{(n)} \notin Z^*_{s_0}$, then $c(s_0, z^{(n)}) = \infty$ and $l(s_0, z^{(n)}) = 0$. We extend the same techniques and notation we used previously to analyze the performance of this scheme. Using the length of $\mathcal{J}_{s_0}(z^{(n)})$, we revise (7) and (8) to:

$$P(y^{(m(n))}) \leq \frac{v_{S[s_0,z^{(n)}]}}{v_{s_0}}\, 2^{-C\cdot c(s_0,z^{(n)})} \tag{35}$$

$$c(s_0, z^{(n)}) \leq \frac{1}{C} I(y^{(m(n))}) + \frac{1}{C} \log_2 \left(\frac{v_{S[s_0,z^{(n)}]}}{v_{s_0}} \right) \tag{36}$$

(9) and (10) remain valid. Modifying equation (11) to account for the length of $\mathcal{J}_{s_0}(z^{(n)})$ yields

$$E[\log_2(D(x))] = -C \cdot c(s_0, z^{(n)}) - \log_2(2e\frac{v_{s_0}}{v_{S[s_0,z^{(n)}]}}), \tag{37}$$

and so

$$c(s_0, z^{(n)}) \geq \frac{1}{C}E[I(y^{(m(n)-1)} \mid z^{(n)})] - \frac{1}{C}\log_2(2e\frac{v_{s_0}}{v_{S[s_0,z^{(n)}]}}). \tag{38}$$

(38) and (13) imply that

$$\begin{aligned} c(s_0, z^{(n)}) &\geq \frac{1}{C}E[I(y^{(m(n))} \mid z^{(n)})] - \frac{1}{C}\log_2\left(\frac{2ev_{s_0}}{p_{min}v_{S[s_0,z^{(n)}]}}\right) \\ &\geq \frac{1}{C}E[I(y^{(m(n))} \mid z^{(n)})] - \frac{1}{C}\log_2\left(\frac{2ev^*}{p_{min}}\right) \end{aligned} \tag{39}$$

where

$$v^* = \max_{i,j\in\{0,\ldots,R-1\}} \frac{v_i}{v_j}. \tag{40}$$

From (39) and (36), we have that the encoder generates cost, on the average, with the ideal of $\frac{H}{C}$ per source symbol; as with the special case of memoryless cost channels, we note that there is a slight deficit in the cost of each code string that is produced and that this deficiency becomes increasingly insignificant as we average over longer and longer source strings.

To analyze the delay between the generation of channel letters at the decoder and the generation of decoded source symbols, we exploit the same ideas and notation that we used earlier in studying memoryless cost channels. Using the length of $\mathcal{J}_{s_0}(z^{(n(m)-1)})$ and letting $c_{max} = \max_{c_{s,j}<\infty} c_{s,j}$, a very similar derivation to those above imply that (15) is revised to

$$E[c(s_0, z^{(n(m))}) \mid y^{(m)}] \leq \frac{1}{C}I(y^{(m)}) + \frac{1}{C}\log_2\left(\frac{v_{S[s_0,z^{(n(m)-1)}]}}{v_{s_0}} \cdot \frac{2e}{2^{-Cc_{max}}}\right). \tag{41}$$

Combining (41) and (39), we find that the extended source sequence $y^{(m')}$ required to produce $z^{(n(m))}$ satisfies (from (39))

$$c(s_0, z^{(n(m))}) \geq \frac{1}{C}E[I(y^{(m')} \mid z^{(n(m))})] - \frac{1}{C}\log_2\left(\frac{2ev_{s_0}}{p_{min}v_{S[s_0,z^{(n(m))}]}}\right) \tag{42}$$

Taking the expected value of both sides of (42) over $z^{(n(m))}$, we find that

$$E[I(y^{(m')} \mid y^{(m)})] - I(y^{(m)}) \leq \log_2 \left(\frac{4e^2 v^*}{p_{min} \cdot 2^{-Cc_{max}}} \right). \tag{43}$$

Therefore, a bound on the number of letters $m' - m$ is

$$E[m' - m \mid y^{(m)}] \leq \frac{\log_2 \left(\frac{4e^2 v^*}{p_{min} \cdot 2^{-Cc_{max}}} \right)}{\log_2 \left(\frac{1}{p_{max}} \right)}. \tag{44}$$

In Appendix I, we modify this analysis to produce an upper bound on the moment generating function for the delay distribution and derive the same bound on the tail of the distribution that we mentioned before for the special case of memoryless cost channels.

Implementation

We now return to the implementation scheme we discussed earlier under the assumption that arithmetic is being performed using binary fixed point arithmetic with M binary digits of accuracy. We recall the two part arithmetic coder illustrated by Figure 4. The outer coder remains unchanged and the inner coder again functions independently of the outer coder. There are a few revisions needed to the earlier description of the inner coder. Instead of producing a bijective mapping of channel sequences into binary sequences, we will create a one-to-one onto mapping of a subset T of channel sequences to the set of binary sequences. Here $T = \{z : z^{(n)} \in Z^*_{s_0}$ for all $n\}$; i.e., T is the set of channel sequences whose initial strings all have finite cost of transmission.

As we saw earlier, when the channel is in state s, we associate each channel letter j with a probability $\frac{v_{S[s,j]}}{v_s} 2^{-Cc_{s,j}}$ and a next state $S[s,j]$. By updating the state after each channel letter is read in and using the appropriate set of channel letter probabilities at each step in the encoding process, the mapping of channel strings into subintervals of the unit interval is essentially the same procedure as the mapping of source strings into subintervals. We can again take advantage of that idea here. We saw that the outer coder created a one-to-one onto mapping of source strings into subintervals. We can modify the outer encoder algorithm to obtain a mapping from T to the set of binary sequences as follows. If the channel is currently in

state s, use the set of probabilities $\{\frac{v_{S[s,j]}}{v_s} 2^{-Cc_{s,j}},\ j \in \{0,\ldots,N-1\}\}$ instead of $\{p_0,\ldots,p_{K-1}\}$ and if the input is channel letter i, update the state of the channel to $S[s,i]$ in order to calculate the next set of channel letter probabilities. The reason why this creates a one-to-one onto mapping from T to the set of binary sequences is that any channel sequence is not in T if and only if it has an initial string corresponding to a subinterval of length zero. The comments made earlier pertaining to the duality between the inner and outer coder given a memoryless cost channel are also applicable here.

Finally, we note that there are no new complications in dealing with sources with memory or adaptive sources. In this case, the encoder and the replica of the encoder at the decoder use $P(y_m \mid y^{(m-1)})$ in place of $P[y_m]$. The source is modeled such that a $p_{min} > 0$ and a $p_{max} < 1$ exist for which none of these probabilities are contained in the open intervals $(0, p_{min})$ and $(p_{max}, 1)$.

CONCLUSION

We have demonstrated that when arithmetic calculations can be accomplished with infinite precision, arithmetic coding encodes exactly at the entropy rate with a delay whose expected value is bounded for a very large class of sources and channels. We have also provided and discussed a simple implementation scheme that focuses on delay under the more realistic assumption that arithmetic is performed using binary fixed point arithmetic with a finite number of degrees of accuracy.

Bibliography

Abramson, N. (1963) *Information Theory and Coding,* McGraw-Hill Book Co., Inc., New York.

Csiszár, I. (1969) "Simple proofs of some theorems on noiseless channels," *Inform. Control* 14, 285-298

Gallager, R. G. (1991) Class Notes for 6.262

Gallager, R. G. (1991) Class Notes for 6.441

Guazzo, M. (1980) "A general minimum-redundancy source-coding algorithm," *I.E.E.E. Trans. Inform. Theory* IT-26, 15-25

Jelinek, F. (1968) *Probabilistic Information Theory,* McGraw-Hill Book Co., Inc., New York.

Jones, C. B. (1981) "An efficient coding system for long source sequences," *I.E.E.E. Trans. Inform. Theory* IT-27, 280-291

Langdon, G. G. (1984) "An introduction to arithmetic coding," *I.B.M. J. Res. Develop.* 28, 135-149

Pasco, R. C. (1976) "Source coding algorithms for fast data compression," Ph.D. thesis, Dept. of Electrical Engineering, Stanford University, CA

Rissanen, J. J. (1976) "Generalized Kraft inequality and arithmetic coding," *I.B.M. J. Res. Develop.* 20, 198-203

Rissanen, J. and G. G. Langdon, Jr. (1979) "Arithmetic coding," *I.B.M. J. Res. Develop.* 23, 149-162

Savari, S. A. (1991) "Source coding for channels with finite-state letter costs," M.S. thesis, Dept. of E.E.C.S., M.I.T., Cambridge, MA

Savari, S. A. and R. G. Gallager (1992) "Arithmetic coding for memoryless cost channels," *Proceedings of the 1992 Data Compression Conference,* Snowbird, Utah.

Shannon, C. E. (1948) "A mathematical theory of communication," *Bell System Tech. J.* 27, 379-423, 623-656

Todd, S. J. P., G. G. Langdon, Jr. and G. N. N. Martin (1983) "A general fixed rate arithmetic coding method for constrained channels," *I.B.M. J. Res. Develop.* 27, 107-115

Witten, I. H., R. M. Neal, and J. G. Cleary (1987) "Arithmetic coding for data compression," *Comm. A.C.M.* 30, 520-540

Appendix I

Memoryless Cost Channels

We maintain the notation developed earlier. Since e^x is a monotonically increasing function of x, inequality (9) implies that

$$E[e^{t \cdot I(y^{(m(n)-1)})} \mid z^{(n)}] \leq E[e^{-t \cdot \log_2(D(x))} \mid z^{(n)}], \; t \geq 0. \tag{45}$$

Since $D(x)$ is uniformly distributed between 0 and $\frac{1}{2} \cdot 2^{-C \cdot [cost\ of\ z^{(n)}]}$, we have that

$$E[e^{-t \cdot \log_2(D(x))} \mid z^{(n)}] = \begin{cases} \frac{e^{t[C \cdot c(z^{(n)})+1]}}{1-t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{46}$$

We also note that

$$e^{t \cdot I(y_{m(n)})} \leq e^{t \cdot \log_2 \left(\frac{1}{p_{min}}\right)} \text{ for all } t \geq 0 \text{ and } y_{m(n)}. \tag{47}$$

Combining (45), (46) and (47), we see that

$$E[e^{t \cdot I(y^{(m(n))})} \mid z^{(n)}] \leq \begin{cases} \frac{e^{t[C \cdot c(z^{(n)})]} \cdot e^{t \cdot \log_2 \left(\frac{2}{p_{min}}\right)}}{1 - t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{48}$$

We can use a very similar argument to bound the moment generating function of the delay between the generation of channel letters at the decoder and the generation of decoded source symbols. The counterpart of the incorporation of (46) into (45) is

$$E[e^{t[C \cdot c(z^{(n(m)-1)})]} \mid y^{(m)}] \leq \begin{cases} \frac{e^{t[I(y^{(m)})+1]}}{1 - t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{49}$$

The analogue of (47) is

$$e^{t[C \cdot c(z_{n(m)})]} \leq e^{t[C c_{max}]} \text{ for all } t \geq 0 \text{ and } z_{n(m)}. \tag{50}$$

(49) and (50) imply

$$E[e^{t[C \cdot c(z^{(n(m))})]} \mid y^{(m)}] \leq \begin{cases} \frac{e^{t[I(y^{(m)})]} \cdot e^{t \cdot \log_2 \left(\frac{2}{2 - C c_{max}}\right)}}{1 - t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{51}$$

Rewriting (48) in terms of m' and $n(m)$ gives

$$E[e^{t \cdot I(y^{(m')})} \mid z^{(n(m))}] \leq \begin{cases} \frac{e^{t[C \cdot c(z^{(n(m))})]} \cdot e^{t \cdot \log_2 \left(\frac{2}{p_{min}}\right)}}{1 - t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{52}$$

From (51) and (52), we see that for any given $y^{(m)}$, the moment generating function of the

self-information of $y^{(m')}$ satisfies

$$E[e^{t[I(y^{(m')} \mid y^{(m)}) - I(y^{(m)})]}] \leq \begin{cases} \frac{e^{t \cdot \log_2\left(\frac{4}{p_{min} \cdot 2^{-C c_{max}}}\right)}}{(1 - t \cdot \log_2 e)^2}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{53}$$

We remarked earlier that

$$(m' - m) \cdot \log_2\left(\frac{1}{p_{max}}\right) \leq I(y^{(m')} \mid y^{(m)}) - I(y^{(m)}) \tag{54}$$

These last two inequalities, combined with a change of variables, imply that

$$E[e^{t(m'-m)}] \leq \begin{cases} \frac{e^{t\left(\frac{\log_2\left(\frac{4}{p_{min} \cdot 2^{-C c_{max}}}\right)}{\log_2\left(\frac{1}{p_{max}}\right)}\right)}}{\left(1 - \frac{t}{\log_e\left(\frac{1}{p_{max}}\right)}\right)^2}, & 0 \leq t < \log_e\left(\frac{1}{p_{max}}\right) \\ \infty, & t \geq \log_e\left(\frac{1}{p_{max}}\right) \end{cases} \tag{55}$$

The Chernoff bound for non-negative random variables is

$$P(Z \geq k) \leq E[e^{tZ}] \cdot e^{-tk} \text{ for all } t \geq 0. \tag{56}$$

Using the Chernoff bound of the distribution of decoding delay gives

$$P(m' - m \geq k) \leq \min_{t \in [0, \log_e(\frac{1}{p_{max}}))} \frac{e^{t\left(\frac{\log_2\left(\frac{4}{p_{min} \cdot 2^{-C c_{max}}}\right)}{\log_2\left(\frac{1}{p_{max}}\right)} - k\right)}}{\left(1 - \frac{t}{\log_e\left(\frac{1}{p_{max}}\right)}\right)^2} \tag{57}$$

It is straightforward to show that for $k > \frac{\log_2\left(\frac{4}{p_{min} \cdot 2^{-C c_{max}}}\right)}{\log_2\left(\frac{1}{p_{max}}\right)} + \frac{2}{\log_e\left(\frac{1}{p_{max}}\right)}$, the minimum of the right hand side of (57) occurs at

$$t = \log_e\left(\frac{1}{p_{max}}\right) - \frac{2}{k - \frac{\log_2\left(\frac{4}{p_{min} \cdot 2^{-C c_{max}}}\right)}{\log_2\left(\frac{1}{p_{max}}\right)}} \tag{58}$$

which corresponds to the bound

$$P(m' - m \geq k) \leq \frac{e^2}{p_{min} \cdot 2^{-Cc_{max}}} \cdot \left(k \cdot \log_e \left(\frac{1}{p_{min}} \right) - \log_e \left(\frac{4}{p_{min} \cdot 2^{-Cc_{max}}} \right) \right)^2 \cdot p_{max}^k. \quad (59)$$

Hence, there exists $\mathcal{K}$ such that for all nonnegative integers k,

$$P(m' - m \geq k) \leq \mathcal{K} \cdot k^2 \cdot p_{max}^k \quad (60)$$

and this indicates that the tail of the distribution is approaching zero at least exponentially.

Finite-State Channels

We generalize the above techniques and notation to handle finite-state channels. (45), (47), (50), and (54) remain valid. Recall that the length of $\mathcal{J}_{s_0}(z^{(n)})$ is $\frac{v_{S[s_0,z^{(n)}]}}{v_{s_0}} 2^{-C \cdot c(s_0,z^{(n)})}$. We then revise (46) and (48) to

$$E[e^{-t \cdot \log_2(D(x))} \mid z^{(n)}] = \begin{cases} \frac{e^{t \cdot \log_2 \left(\frac{v_{s_0}}{v_{S[s_0,z^{(n)}]}} 2^{C \cdot c(s_0,z^{(n)})+1} \right)}}{1 - t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \quad (61)$$

and

$$E[e^{t \cdot I(y^{(m(n))})} \mid z^{(n)}] \leq \begin{cases} \frac{e^{t \cdot \log_2 \left(\frac{v_{s_0}}{v_{S[s_0,z^{(n)}]}} 2^{C \cdot c(s_0,z^{(n)})} \right)} \cdot e^{t \cdot \log_2 \left(\frac{2}{p_{min}} \right)}}{1 - t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \quad (62)$$

respectively. Likewise, the generalization of (49) is

$$E\left(e^{t \left(C \cdot c(z^{(n(m)-1)}) - \log_2 \left(\frac{v_{S[s_0,z^{(n(m)-1)}]}}{v_{s_0}} \right) \right)} \mid y^{(m)} \right) \leq \begin{cases} \frac{e^{t[I(y^{(m)})+1]}}{1 - t \cdot \log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \quad (63)$$

Using (63) and (50), we revise (51) to

$$E[e^{t[C\cdot c(z^{(n(m))})]} \mid y^{(m)}] \leq \begin{cases} \frac{e^{t[I(y^{(m)})]} \cdot e^{t\cdot\log_2\left(2\cdot 2^{C c_{max}}\cdot \frac{v_{S[s_0,z(n(m)-1)]}}{v_{s_0}}\right)}}{1-t\cdot\log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{64}$$

If we rewrite (62) in terms of m' and $n(m)$, we find that

$$E[e^{t\cdot I(y^{(m')})} \mid z^{(n(m))}] \leq \begin{cases} \frac{e^{t\cdot\log_2\left(\frac{v_{s_0}}{v_{S[s_0,z^{(n(m))}]}}\, 2^{C\cdot c(s_0,z^{(n(m))})}\right)} \cdot e^{t\cdot\log_2\left(\frac{2}{p_{min}}\right)}}{1-t\cdot\log_2 e}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{65}$$

(64) and (65) imply that for any given $y^{(m)}$, the moment generating function of the self-information of $y^{(m')}$ satisfies

$$E[e^{t[I(y^{(m')} \mid y^{(m)}) \,-\, I(y^{(m)})]} \leq \begin{cases} \frac{e^{t\cdot\log_2\left(\frac{4v^*}{p_{min}\cdot 2^{-C c_{max}}}\right)}}{(1-t\cdot\log_2 e)^2}, & 0 \leq t < \log_e 2 \\ \infty, & t \geq \log_e 2 \end{cases} \tag{66}$$

By comparing (66) to (53), we see that we can generalize (55), (57), (58) and (59), by replacing $\frac{4}{p_{min}\cdot 2^{-C c_{max}}}$ everywhere, including the condition on k, with $\frac{4v^*}{p_{min}\cdot 2^{-C c_{max}}}$. Clearly, for appropriate $\mathcal{K}$, (60) remains valid.

Appendix II

The notation developed earlier is retained. For this analysis, we assume that arithmetic calculations are performed with accuracy to M binary digits and with nearest point round-off. Recall that $\mathcal{I}_{norm}(y^{(n)})$ is the interval after the n^{th} symbol is read in, the special round-off rules for endpoints in the interval $[\frac{1}{2} - 2^{-L}, \frac{1}{2})$ are applied, and renormalization has taken place. We let $\tilde{\mathcal{I}}(y^{(n)})$ denote the interval after the n^{th} letter is inputted and the special round-off rules are executed, but before renormalization. Then

$$\log_2\left(\frac{|\mathcal{I}_{norm}(y^{(n)})|}{|\tilde{\mathcal{I}}(y^{(n)})|}\right) = \text{ number of binary digits emitted from the encoder at time } n.$$

If we define $Q[y^{(n)}] = \dfrac{|\tilde{\mathcal{I}}(y^{(n)})|}{|\mathcal{I}_{norm}(y^{(n-1)})|}$, then $Q[y^{(n)}]$ is the shrinkage factor corresponding to y_n and is therefore a probability on $\{y_n\}$ for any given $y^{(n-1)}$. We have

$$\begin{aligned}\sum_{n=1}^{m} -\log_2 Q[y^{(n)}] &= \sum_{n=1}^{m} -\log_2 \left(\frac{|\tilde{\mathcal{I}}(y^{(n)})|}{|\mathcal{I}_{norm}(y^{(n)})|} \cdot \frac{|\mathcal{I}_{norm}(y^{(n)})|}{|\mathcal{I}_{norm}(y^{(n-1)})|} \right) \\ &= -\log_2 |\mathcal{I}_{norm}(y^{(m)})| + \text{Total number of bits out of the encoder by time } m \end{aligned} \quad (67)$$

We have that

$$|\mathcal{I}_{norm}(y^{(m)})| > 2^{-L} \quad (68)$$

for all m and $y^{(m)}$ because the right endpoint of $|\mathcal{I}_{norm}(y^{(m)})|$ is greater that $\frac{1}{2}$ and the left endpoint of $\tilde{\mathcal{I}}(y^{(m)})$ cannot lie in $[\frac{1}{2}-2^{-L}, \frac{1}{2})$ since the special round-off rule would then eliminate straddling. Hence,

$$\lim_{m\to\infty} \frac{\sum_{n=1}^{m} -\log_2 Q[y^{(n)}]}{m} = \lim_{m\to\infty} \frac{\text{Total number of bits emitted after processing } y^{(m)}}{m} \quad (69)$$

Lemma 1 *For all m and $y^{(m)}$, if binary digits are emitted at time m, then*

$$|\mathcal{I}_{norm}(y^{(m)})| \geq \frac{[p_{min} - 2 \cdot 2^{L-M}]}{2} \quad (70)$$

Proof: We use induction. As basis, for $m = 0$, $|\mathcal{I}_{norm}(y^{(m)})| = 1$. Now assume the lemma is true for $m-1$ and establish it for m. Assume that binary digits are emitted at time m, so that $\tilde{\mathcal{I}}(y^{(m)})$ doesn't straddle $\frac{1}{2}$. Let $\frac{1}{2} + z$ be the midpoint of the smallest binary interval containing $\tilde{\mathcal{I}}(y^{(m)})$. Then $\frac{1}{2} + z \in \tilde{\mathcal{I}}(y^{(m)}) \subset \mathcal{I}_{norm}(y^{(m-1)})$. Since $\frac{1}{2} \in \mathcal{I}_{norm}(y^{(m-1)})$, $|\mathcal{I}_{norm}(y^{(m-1)})| \geq |z|$.

Case 1: The special round-off rule doesn't move the left endpoint of $\tilde{\mathcal{I}}(y^{(m)})$. Then

$$\begin{aligned}|\tilde{\mathcal{I}}(y^{(m)})| &\geq |\mathcal{I}_{norm}(y^{(m-1)})| \cdot (p_{min} - 2^{-M}) - 2^{-M}, \\ &\qquad \text{since we use nearest endpoint round-off to calculate both endpoints of } \tilde{\mathcal{I}}(y^{(m)}). \\ &\geq |\mathcal{I}_{norm}(y^{(m-1)})| \cdot p_{min} - 2 \cdot 2^{-M}, \text{ since } |\mathcal{I}_{norm}(y^{(m-1)})| \leq 1 \\ &= |\mathcal{I}_{norm}(y^{(m-1)})| \cdot \left(p_{min} - \frac{2 \cdot 2^{-M}}{|\mathcal{I}_{norm}(y^{(m-1)})|} \right)\end{aligned}$$

$$\geq |\mathcal{I}_{norm}(y^{(m-1)})| \cdot [p_{min} - 2 \cdot 2^{L-M}], \text{ since } |\mathcal{I}_{norm}(y^{(m-1)})| > 2^{-L}$$
$$\geq |z| \cdot [p_{min} - 2 \cdot 2^{L-M}].$$

Case 2: The special round-off rule moves the left endpoint of $\tilde{\mathcal{I}}(y^{(m)})$. Then $|\tilde{\mathcal{I}}(y^{(m)})| \geq |z| > |z| \cdot [p_{min} - 2 \cdot 2^{L-M}]$.

In both cases, the size of the smallest binary interval containing $\tilde{\mathcal{I}}(y^{(m)})$ is upper bounded by $2|z|$. Hence,

$$\mathcal{I}_{norm}(y^{(m)}) \geq \frac{1}{2|z|}|\tilde{\mathcal{I}}(y^{(m)})| \geq \frac{1}{2}[p_{min} - 2 \cdot 2^{L-M}]. \square$$

If we denote the probability of the n^{th} symbol in the source sequence by $P[y_n]$, then, from (69), the redundancy of the encoder is

$$R = \lim_{m \to \infty} E\left(\frac{1}{m}\sum_{n=1}^{m} \log_2\left(\frac{P[y_n]}{Q[y^{(n)}]}\right)\right), \tag{71}$$

where the expectation is over all source sequences y.

Let y_k^l denote the source string $\{y_k, y_{k+1}, \ldots, y_l\}$ for $l \geq k$. Suppose we parse our source sequence y into $\{y_1^{i_1}, y_{i_1+1}^{i_2}, y_{i_2+1}^{i_3} \cdots\}$ where $i_1, i_2, i_3, \ldots$ are the (random) times at which binary digits come out of the encoder. Define

$$R_k = E\left(\sum_{n=i_k+1}^{i_{k+1}} \log_2\left(\frac{P[y_n]}{Q[y^{(n)}]}\right)\right), \tag{72}$$

where the expectation is over all source sequences y which have $y^{(i_k)}$ as a prefix. We will establish an upper bound $\overline{R}$ on R_k that is independent of $y^{(i_k)}$; from (71) and (72), it is clear that $R \leq \overline{R}$. Let $\mathcal{I}_0 = \mathcal{I}_{norm}(y^{(i_k)})$; from Lemma 1, $|\mathcal{I}_0| \geq \frac{1}{2}[p_{min} - 2 \cdot 2^{L-M}]$. Let $y_1', y_2', \ldots$ be the values of $y_{i_k+1}, y_{i_k+2}, \ldots$ that cause straddling and let $p_1', p_2', \ldots$ be the probabilites of these letters, respectively (these letters and probabilities are functions of $y^{(i_k)}$). Let $\mathcal{I}_1$, $\mathcal{I}_2$, ... be the corresponding intervals.

Lemma 2 *For* $0 \leq l < i_{k+1} - i_k$,

$$|\mathcal{I}_l| \geq |\mathcal{I}_0| \cdot \prod_{j=1}^{l} p_j' - \frac{2 \cdot 2^{-M}}{1 - p_{max}}. \tag{73}$$

Proof: Since no renormalization occurs after symbols $y_{i_k+1}, \ldots, y_{i_k+l}$ are processed, we see that

$$\begin{aligned}
|\mathcal{I}_l| &\geq |\mathcal{I}_{l-1}| \cdot (p_l' - 2^{-M}) - 2^{-M} \\
&\geq |\mathcal{I}_{l-1}| \cdot p_l' - 2 \cdot 2^{-M} \\
&\geq [|\mathcal{I}_{l-2}| \cdot p_{l-1}' - 2 \cdot 2^{-M}] \cdot p_l' - 2 \cdot 2^{-M} \\
&\geq \cdots \\
&\geq |\mathcal{I}_0| \cdot p_1' p_2' \cdots p_l' - 2 \cdot 2^{-M}[1 + p_l' + p_l' p_{l-1}' + \cdots] \\
&\geq |\mathcal{I}_0| \cdot \prod_{j=1}^{l} p_j' - \frac{2 \cdot 2^{-M}}{1 - p_{max}} \text{ since } p_j' \leq p_{max} \text{ for all } j. \square
\end{aligned}$$

Now let $\mathcal{L} = i_{k+1} - i_k$. Then $P(\mathcal{L} \geq l) = p_1' p_2' \cdots p_{l-1}'$ assuming that the special round-off rule doesn't force renormalization before l. Let l_s be the maximum value of $\mathcal{L}$ for a given $\mathcal{I}_{norm}(y^{(i_k)})$. Then

$$R_k = \sum_{l=1}^{l_s} p_1' p_2' \cdots p_{l-1}' E_{y_{i_k+l}} \left(\log_2 \left(\frac{P[y_{i_k+l}]}{Q[y^{(i_k)}, y_1', \ldots, y_{l-1}', y_{i_k+l}]} \right) \right) \tag{74}$$

The quantity in brackets, conditional on $y^{(i_k+l-1)}$, is simply a divergence.

Lemma 3 *For arbitrary probability assignments* $\{p_i\}$ *and* $\{q_i\}$, *if* $\epsilon_i = p_i - q_i$ *for* $i = 1, \ldots, K$, *then*

$$\sum_{i=1}^{K} p_i \log_e \left(\frac{p_i}{q_i} \right) \leq \sum_{i=1}^{K} \frac{\epsilon_i^2}{2(p_i - |\epsilon_i|)}$$

Proof: We have

$$\begin{aligned}
\sum_{i=1}^{K} p_i \log_e \left(\frac{p_i}{q_i} \right) &= \sum_{i=1}^{K} p_i \log_e \left(\frac{p_i}{p_i - \epsilon_i} \right) \\
&= \sum_{i=1}^{K} -p_i \log_e \left(1 - \frac{\epsilon_i}{p_i} \right)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{K} p_i \sum_{n=1}^{\infty} \frac{\left(\frac{\epsilon_i}{p_i}\right)^n}{n} \\
&= \sum_{i=1}^{K} \sum_{n=2}^{\infty} p_i \frac{\left(\frac{\epsilon_i}{p_i}\right)^n}{n}, \text{ since } \sum_{i=1}^{K} \epsilon_i = 0 \\
&\leq \sum_{i=1}^{K} p_i \cdot \frac{\epsilon_i^2}{2p_i^2} \cdot \left(1 + \frac{|\epsilon_i|}{p_i} + \frac{|\epsilon_i|^2}{p_i^2} + \cdots\right) \\
&\leq \sum_{i=1}^{K} p_i \cdot \frac{\epsilon_i^2}{2p_i^2} \cdot \frac{1}{1 - \frac{|\epsilon_i|}{p_i}} \\
&\leq \sum_{i=1}^{K} \frac{\epsilon_i^2}{2(p_i - |\epsilon_i|)} \quad \square
\end{aligned}
$$

Now consider the terms in (74) with $l < l_s$. In these cases, the special round-off rules are not in effect. Let $\{q_i\}$ be given by $q_i = Q[y^{(i_k)}, y_1', \ldots, y_{l-1}', i]$. Then

$$
\begin{aligned}
q_i \cdot |\mathcal{I}_{l-1}| &\geq |\mathcal{I}_{l-1}| \cdot (p_i - 2^{-M}) - 2^{-M} \geq p_i \cdot |\mathcal{I}_{l-1}| - 2 \cdot 2^{-M} \qquad (75) \\
&\leq |\mathcal{I}_{l-1}| \cdot (p_i + 2^{-M}) + 2^{-M} \leq p_i \cdot |\mathcal{I}_{l-1}| + 2 \cdot 2^{-M} \qquad (76)
\end{aligned}
$$

Hence, $|\epsilon_i| \cdot |\mathcal{I}_{l-1}| \leq 2 \cdot 2^{-M}$. Using (73),

$$
|\epsilon_i| \leq \frac{2 \cdot 2^{-M}}{|\mathcal{I}_0| \cdot \prod_{j=1}^{l-1} p_j' - \frac{2 \cdot 2^{-M}}{1 - p_{max}}} \qquad (77)
$$

Also, from (68), $|\epsilon_i| \leq \frac{2 \cdot 2^{-M}}{|\mathcal{I}_{l-1}|} \leq \frac{2 \cdot 2^{-M}}{2^{-L}} = 2 \cdot 2^{L-M}$. Therefore,

$$
\begin{aligned}
&E_{y_{i_k+l}}\left(\log_2\left(\frac{P[y_{i_k+l}]}{Q[y^{(i_k)}, y_1', \ldots, y_{l-1}', y_{i_k+l}]}\right)\right) \leq \\
&\frac{2K \log_2 e}{2^{2M} \cdot \left(|\mathcal{I}_0| \cdot \prod_{i=1}^{l-1} p_i' - \frac{2 \cdot 2^{-M}}{1 - p_{max}}\right)^2 \cdot (p_{min} - 2 \cdot 2^{L-M})} \qquad (78)
\end{aligned}
$$

Next, we examine the term in (74) with $l = l_s$. Let $\mathcal{I}^*(y^{(n)})$ denote the interval after the n^{th} letter is read in, but before the special round-off rule is applied. If we define $R[y^{(n)}] = \frac{|\mathcal{I}^*(y^{(n)})|}{|\mathcal{I}_{norm}(y^{(n-1)})|}$, then $R[y^{(n)}]$ is a probability that represents the shrinkage factor corresponding to y_n before the execution of the special round-off rule. Let $\{r_i\}$ be given by

$r_i = R[y^{(i_k)}, y_1', \ldots, y_{l-1}', i]$. Then

$$E_{y_{i_k+l_s}} \left(\log_2 \left(\frac{P[y_{i_k+l_s}]}{Q[y^{(i_k)}, y_1', \ldots, y_{l_s-1}', y_{i_k+l_s}]} \right) \right) = (\log_2 e) \cdot \left(\sum_{i=1}^{K} p_i \log_e \left(\frac{p_i}{q_i} \right) \right)$$
$$= (\log_2 e) \cdot \left(\sum_{i=1}^{K} p_i \log_e \left(\frac{p_i}{r_i} \right) + \sum_{i=1}^{K} p_i \log_e \left(\frac{r_i}{q_i} \right) \right) \quad (79)$$

We saw earlier that

$$\sum_{i=1}^{K} p_i \log_e \left(\frac{p_i}{r_i} \right) \leq \frac{2K}{2^{2M} \cdot (|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}})^2 \cdot (p_{min} - 2 \cdot 2^{L-M})}. \quad (80)$$

Combining (74), and (78) to (80), we have that

$$R_k \leq R_{k,1} + R_{k,2} \quad (81)$$

where

$$R_{k,1} = \sum_{l=1}^{l_s} \frac{p_1' \ldots p_{l-1}' \cdot 2K \log_2 e}{(2^M \cdot |\mathcal{I}_0| \cdot \prod_{i=1}^{l-1} p_i' - \frac{2}{1-p_{max}})^2 \cdot (p_{min} - 2 \cdot 2^{L-M})} \quad (82)$$

and

$$R_{k,2} = p_1' \ldots p_{l_s-1}' \cdot \sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i} \right). \quad (83)$$

We first bound $R_{k,1}$. Let $\alpha = \frac{2K \log_2 e}{(p_{min} - 2 \cdot 2^{L-M}) \cdot 2^{2M} \cdot |\mathcal{I}_0|}$. Then

$$R_{k,1} = \sum_{l=1}^{l_s} \frac{\alpha(|\mathcal{I}_0| p_1' \ldots p_{l-1}')}{(|\mathcal{I}_0| \cdot \prod_{i=1}^{l-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}})^2}$$
$$= \frac{\alpha(|\mathcal{I}_0| p_1' \ldots p_{l_s-1}')}{(|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}})^2} \sum_{l=1}^{l_s} \left(\frac{|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}}}{|\mathcal{I}_0| \cdot \prod_{i=1}^{l-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}}} \right)^2 \cdot \frac{\prod_{i=1}^{l-1} p_i'}{\prod_{i=1}^{l_s-1} p_i'}$$
$$\leq \left(\frac{\alpha}{|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}}} + \frac{\frac{2 \cdot 2^{-M}}{1-p_{max}} \cdot \alpha}{(|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}})^2} \right) \cdot \sum_{l=1}^{l_s} \left(\prod_{i=l}^{l_s-1} p_i' \right)^2 \cdot \left(\prod_{i=l}^{l_s-1} p_i' \right)^{-1}$$
$$\leq \left(\frac{\alpha}{|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}}} + \frac{\frac{2 \cdot 2^{-M}}{1-p_{max}} \cdot \alpha}{(|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}})^2} \right) \cdot (1 + p_{l_s-1}' + p_{l_s-1}' p_{l_s-2}' + \ldots)$$

$$\leq \left(\frac{\alpha}{|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}}} + \frac{\frac{2 \cdot 2^{-M}}{1-p_{max}} \cdot \alpha}{(|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}})^2} \right) \cdot \frac{1}{1-p_{max}} \text{ since } p_i' \leq p_{max} \quad (84)$$

The reasoning in Lemma 2 can be extended to show that for $0 \leq l < i_{k+1} - i_k$,

$$|\mathcal{I}_l| \leq |\mathcal{I}_0| \cdot p_1' \ldots p_l' + \frac{2 \cdot 2^{-M}}{1-p_{max}} \quad (85)$$

(85) and (68) imply that

$$|\mathcal{I}_0| \cdot \prod_{i=1}^{l_s-1} p_i' - \frac{2 \cdot 2^{-M}}{1-p_{max}} \geq |\mathcal{I}_{l_s-1}| - \frac{4 \cdot 2^{-M}}{1-p_{max}} \geq 2^{-L} - \frac{4 \cdot 2^{-M}}{1-p_{max}} \quad (86)$$

Substituting (86) and the value of α into (84) and letting $\gamma = 2 \cdot 2^{L-M}$, we find that

$$R_{k,1} \leq \frac{K \log_2 e \cdot \gamma \cdot (1 - p_{max} - \gamma)}{|\mathcal{I}_0| \cdot (p_{min} - \gamma) \cdot (1 - p_{max} - 2\gamma)^2} \cdot 2^{-M} \quad (87)$$

We now consider $R_{k,2}$. To evaluate $\sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i}\right)$, we note that the special round-off rule changes the lengths of either zero or two intervals. If the special round-off rule does not alter the length of any interval, then $r_i = q_i$ for all i and $\sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i}\right) = 0$. Otherwise, there is some j for which

$$\sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i}\right) = p_j \log_2 \left(\frac{r_j}{q_j}\right) + p_{j+1} \log_2 \left(\frac{r_{j+1}}{q_{j+1}}\right).$$

By referring to figure 5, it is clear that $r_j + r_{j+1} = q_j + q_{j+1}$; hence, for some non-negative x, $q_j = r_j + x$ and $q_{j+1} = r_{j+1} - x$. We are therefore interested in bounding

$$\sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i}\right) = p_j \log_2 \left(\frac{r_j}{r_j + x}\right) + p_{j+1} \log_2 \left(\frac{r_{j+1}}{r_{j+1} - x}\right) < p_{j+1} \log_2 \left(\frac{r_{j+1}}{r_{j+1} - x}\right). \quad (88)$$

(75) provides an upper bound on $p_{j+1} - r_{j+1}$. If we incorporate this bound into (88), we see that

$$\sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i}\right) \leq \left(r_{j+1} + \frac{2 \cdot 2^{-M}}{|\mathcal{I}_{l_s-1}|}\right) \log_2 \left(\frac{r_{j+1}}{r_{j+1} - x}\right). \quad (89)$$

By taking partial derivatives of the above expression with respect to x and r_{j+1}, we can demonstrate that the right-hand side of (89) is maximized when x takes on its largest possible value and $r_{j+1} - x$ assumes its smallest possible value. Hence, $\sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i}\right)$ is maximized when the left and right endpoints of $\mathcal{I}^*(y^{(i_k)}, y'_1, \ldots, y'_{l_s-1}, j+1)$ are $\frac{1}{2} - 2^{-L}$ and $\frac{1}{2} + 2^{-M}$, respectively; i.e., when

$$x = \frac{2^{-L}}{|\mathcal{I}_{l_s-1}|}, \tag{90}$$

$$\text{and } r_{j+1} = \frac{1}{|\mathcal{I}_{l_s-1}|}(2^{-L} + 2^{-M}). \tag{91}$$

Hence,

$$\sum_{i=1}^{K} p_i \log_2 \left(\frac{r_i}{q_i}\right) \leq \left(3 + \frac{2}{\gamma}\right) \log_2 \left(1 + \frac{2}{\gamma}\right) \cdot \frac{2^{-M}}{|\mathcal{I}_{l_s-1}|} \tag{92}$$

and so (86), (75) and (68) imply that

$$R_{k,2} \leq \frac{\prod_{i=1}^{l_s-1} p'_i}{|\mathcal{I}_{l_s-1}|} \cdot \left(3 + \frac{2}{\gamma}\right) \log_2 \left(1 + \frac{1}{\gamma}\right) \cdot 2^{-M} \leq \left(1 + \frac{\gamma}{1 - p_{max}}\right) \cdot \left(3 + \frac{2}{\gamma}\right) \log_2 \left(1 + \frac{1}{\gamma}\right) \cdot \frac{2^{-M}}{|\mathcal{I}_0|} \tag{93}$$

If we combine the results of (87) and (93) and maximize over $|\mathcal{I}_0|$ in the range $\frac{p_{min}-\gamma}{2} = \frac{p_{min}-2\cdot 2^{L-M}}{2} \leq |\mathcal{I}_0| \leq 1$, we find that the maximum is achieved at $|\mathcal{I}_0| = \frac{p_{min}-\gamma}{2}$ and therefore

$$\begin{aligned} R_k \leq \overline{R} \leq & \frac{2K \log_2 e \cdot \gamma \cdot (1 - p_{max} - \gamma)}{(p_{min} - \gamma)^2 \cdot (1 - p_{max} - 2\gamma)^2} \cdot 2^{-M} \\ & + \frac{2}{p_{min} - \gamma} \cdot \left(1 + \frac{\gamma}{1 - p_{max}}\right) \cdot \left(3 + \frac{2}{\gamma}\right) \log_2 \left(1 + \frac{2}{\gamma}\right) \cdot 2^{-M} \end{aligned} \tag{94}$$

In theory, it is possible to minimize the bound in (94) or an approximation of it over γ between $4 \cdot 2^{-M}$ and $\min\left(p_{min}, \frac{1-p_{max}}{2}\right)$ in order to find an appropriate value of 2^{-L}. For $2^{-L} = \frac{6 \cdot 2^{-M}}{p_{min}}$, i.e., for $\gamma = \frac{p_{min}}{3}$, we have that

$$\overline{R} < \left(\frac{3(1 - p_{max}) - p_{min}}{2p_{min}(1 - p_{max} - \frac{2}{3} \cdot p_{min})^2} + \left(\frac{9}{p_{min}} + \frac{3}{1 - p_{max}}\right) \cdot \left(1 + \frac{2}{p_{min}}\right) \cdot \log_e \left(1 + \frac{6}{p_{min}}\right)\right) \cdot 2^{-M} \tag{95}$$

We have demonstrated that there is some constant Υ for which $\overline{R} \leq \Upsilon \cdot 2^{-M}$. Hence,

$$R \leq \overline{R} \leq \Upsilon \cdot 2^{-M}.$$

In the case of other round-off rules in which numbers are rounded to within 2^{-M} of their value, the above analysis holds with minor modifications in the calculation of constants.

We conclude by noting that this analysis does not rely upon the fact that the source is memoryless. It can be utilized for any source which is modeled so that no letter probability is contained in the open intervals $(0, p_{min})$ and $(p_{max}, 1)$ for some $p_{min} > 0$ and $p_{max} < 1$.