The Sliding-Window Lempel–Ziv Algorithm is Asymptotically Optimal

AARON D. WYNER, FELLOW, IEEE, AND JACOB ZIV, FELLOW, IEEE

Invited Paper

The sliding-window version of the Lempel–Ziv data-compression algorithm (sometimes called LZ '77) has been thrust into prominence recently. A version of this algorithm is used in the highly successful "Stacker" program for personal computers. It is also incorporated into Microsoft's new MS-DOS-6. Although other versions of the Lempel–Ziv algorithm are known to be optimal in the sense that they compress a data source to its entropy, optimality in this sense has never been demonstrated for this version.

In this self-contained paper, we will describe the algorithm, and show that as the "window size," a quantity which is related to the memory and complexity of the procedure, goes to infinity, the compression rate approaches the source entropy. The proof is surprisingly general, applying to all finite-alphabet stationary ergodic sources.

I. Introduction

The sliding-window version of the Lempel–Ziv (LZ) data compression algorithm (first proposed in [6] in 1977 and sometimes called LZ '77) has been thrust into prominence recently. A version of this algorithm is used in the highly successful "Stacker" program for personal computers [3]. It is also incorporated into Microsoft's new MS-DOS-6. Although other versions of the LZ algorithm are known to be optimal in the sense that they compress a data source to its entropy, such optimality has never been demonstrated for the sliding-window version.

This self-contained paper begins with a description of the sliding-window LZ algorithm. The main result is a theorem which asserts that as the "window size" (a quantity directly related to the memory and complexity requirements of the procedure) becomes large, the compression rate approaches the source entropy. This theorem is surprisingly general, applying to all stationary, ergodic, finite-alphabet sources.

We will be concerned with a data source which is a random sequence $\{X_k\}_{k=-\infty}^{\infty}$, that is stationary, ergodic, and takes values in the alphabet $\mathcal{A}$ with cardinality $|\mathcal{A}| = A < \infty$. For $-\infty \leq i < j \leq \infty$, let $\boldsymbol{X}_i^j$ denote the substring $(X_i, X_{i+1}, \cdots, X_j)$. Let

$$\begin{aligned} H_n &= \frac{1}{n} H(\boldsymbol{X}_1^n) \\ &\triangleq -\frac{1}{n} \sum_{\boldsymbol{X} \in \mathcal{A}^n} \Pr\{\boldsymbol{X}_1^n = \boldsymbol{X}\} \log(\Pr\{\boldsymbol{X}_1^n = \boldsymbol{X}\}) \end{aligned}$$

be the nth order (normalized) entropy of $\{X_k\}$, and let $H = \lim_{n\to\infty} H_n$ be the source entropy. (All logarithms in this paper are taken to the base 2.) It is well known that this limit always exists, and that the data source can be losslessly encoded using $(H+\epsilon)$ bits per source symbol [1], [2] (for arbitrarily $\epsilon > 0$). The LZ algorithm is a universal procedure (which does not depend on the source statistics) for encoding the source.

In Section II we give a precise description of the sliding-window LZ algorithm and state the main result. In Section III we establish the mathematical facts that we need in the proof of this theorem which we give in Section IV.

Let us remark at this point that the sliding-window LZ algorithm discussed here is not the same as the "LZ '78" algorithm, used for example in the UNIX "compress" command and in a CCITT standard for data compression for modems. In LZ '78, the "dictionary" is allowed to grow until it reaches a certain specified size. In the sliding-window version, the dictionary is of fixed size, and consists of the data symbols immediately preceding the data being compressed.

Finally, we remark that the treatment in [6], where the sliding-window LZ algorithm is first described, is nonprobabilistic. In that reference, the algorithm is shown to be optimal for a certain (deterministic) family of sources. Furthermore, for a given source in this family, the rate of convergence is similar to that of the best nonuniversal compression scheme. In the present paper, we replace the assumption that the source belongs to this family, by the more conventional (and realistic) assumption that the source is stationary and ergodic.

Manuscript received November 1, 1993; revised January 15, 1994.

A. D. Wyner is with AT&T Bell Laboratories, Murray Hill, NJ 07974, USA.

J. Ziv was with AT&T Bell Laboratories, Murray Hill, NJ 07974, USA. He is now with the Faculty of Electrical Engineering, Technion–Israel Institute of Technology, Haifa, Israel.

IEEE Log Number 9401404.

0018–9219/94$04.00 © 1994 IEEE

II. Description of the Algorithm

Let $\{X_k\}_{k=-\infty}^{\infty}$, be a stationary ergodic random process with entropy H, as discussed in Section I. We will now describe the sliding-window Lempel–Ziv algorithm for encoding the sequence $\{X_k\}_{k=1}^{N}$, where N is a large integer. Later we shall let $N \to \infty$.

Let $n_w > 0$ be an integer parameter, called the *window size.* Assume that n_w is a power of two. The first n_w symbols of $\boldsymbol{X}_1^N$ are encoded with no attempt at compression. We call $\boldsymbol{X}_1^{n_w}$ the *window*. The number of bits required to encode the window $\boldsymbol{X}_1^{n_w}$ is $\lceil n_w \log A \rceil$. These bits are to be considered as overhead that will be ammortized over a very long time. We next define the first *phrase* $\boldsymbol{Y}_1 \triangleq \boldsymbol{X}_{n_w+1}^{n_w+L_1}$, where L_1 is the largest integer such that

$$\boldsymbol{X}_{n_w+1}^{n_w+L_1} = \boldsymbol{X}_{n_w-m}^{n_w-m+L_1-1}, \qquad \text{for some } m \in [0, n_w - 1]. \tag{1}$$

Thus L_1 is the largest integer such that a copy of $\boldsymbol{X}_{n_w+1}^{n_w+L_1}$ begins in the window. Let m_1 be the (say) smallest of those m's which satisfy (1). If (1) is not satisfied for any $m \in [0, n_w - 1]$, that is, $X_{n_w+1} \neq X_m$, for all $m \in [1, n_w]$, then we take $L_1 = 1$. For example if $n_w = 5$, and $(X_1, X_2, \cdots) = (a\,b\,c\,d\,e \vdots d\,e\,d\,a \cdots)$, then $L_1 = 3$ since the string $(d\ e\ d)$ begins at position 4 $(\leq n_w)$, i.e., $\boldsymbol{X}_{n_w+1}^{n_w+3} = \boldsymbol{X}_{n_w-1}^{n_w-1+2}$. Also $m_1 = 1$. Note that there is no upper bound on L_1 and in particular it can exceed n_w—for example, if $n_w = 5$ and $(X_1, X_2, \cdots) = (b\,b\,b\,b\,a \vdots a\,a\,a\,a\,a\,a\,a\,a\,a\,b \cdots)$, then $L_1 = 10$.

We now show how to encode the data sequence $\boldsymbol{Y}_1$. The first part of the encoding is a comma-free binary encoding of L_1. One such encoding is the mapping $e(L)$ given in the Appendix. Immediately following $e(L_1)$ is a binary string s_1, where

$$s_1 = \begin{cases} \text{binary encoding of } m_1, \\ \qquad \qquad \text{if } \log n_w < \lceil L_1 \log A \rceil \\ \text{binary encoding (no compression) of } \boldsymbol{Y}_1, \\ \qquad \qquad \text{if } \log n_w \geq \lceil L_1 \log A \rceil. \end{cases} \tag{2}$$

Assume that $n_w \geq A$, so that when $L_1 = 1, s_1$, is an encoding of $\boldsymbol{Y}_1$. Thus the total number of bits needed to encode the first phrase $\boldsymbol{Y}_1 = \boldsymbol{X}_{n_w+1}^{n_w+L_1}$ is $|e(L_1)| + |s_1|$. Since $m_1 \in [0, n_w - 1]$, its binary encoding requires $\log n_w$ bits. Also $\lceil L_1 \log A \rceil$ bits are required to encode $\boldsymbol{Y}_1$ with no compression. Thus the length of the encoding of $\boldsymbol{Y}_1$ is

$$\begin{aligned} &|e(L_1)| + \min(\log n_w, \lceil L_1 \log A \rceil) \\ &\qquad \leq \min(\log n_w + \gamma_1 \log(L_1 + 1), \gamma_2 L_1) \end{aligned} \tag{3}$$

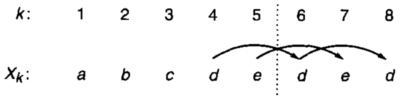
for suitably large γ_1, γ_2. We made use of (A.4b) in (3).

Now observe that with knowledge of the window $\boldsymbol{X}_1^{n_w}, e(L_1)$, and s_1, the decoder can recover $Y_1 = \boldsymbol{X}_{n_w+1}^{n_w+L_1}$. Here is how. The first step is to decode $e(L_1)$ to recover L_1. The decoder now computers $\lceil L_1 \log A \rceil$. If this quantity is $\leq \log n_w$, the next $\rceil L_1 \log A \rceil$ bits are a binary encoding of $\boldsymbol{Y}_1$. Otherwise, the next $\log n_w$ bits define a pointer to position $(n_w - m_1)$ where a copy of $\boldsymbol{Y}_1$ begins. This position is within the window, i.e., $n_w - m_1 \in [1, n_w]$. The decoder copies $X_{n_w-m_1}$ into position $n_w + 1$, then $X_{n_w-m_1+1}$ into position $n_w + 2$, etc. When $X_{n_w-m+L_1-1}$ is copied into position $n_w + L_1$, the process is complete and $\boldsymbol{Y}_1$ is reconstructed.

As an example, consider again the case $n_w = 5$, where

$$(X_1, X_2, \cdots) = (a\,b\,c\,d\,e \vdots d\,e\,d\,a \cdots).$$

As above, $L_1 = 3$ and $m_1 = 1$. With knowledge of the window, $\boldsymbol{X}_1^{n_w} = (a\,b\,c\,d\,e)$, the decoder can copy "$d$" and "$e$" to positions 6 and 7, respectively, and then copy the "d" in position 6 to position 8:



Thus $\boldsymbol{Y}_1 = (d\,e\,d)$.

After the first phrase $\boldsymbol{Y}_1$ is encoded, the first L_1 symbols in the window are deleted and the L_1 symbols of $\boldsymbol{Y}_1$ are added at the end of the window. Thus the new window is $\boldsymbol{X}_{L_1+1}^{L_1+n_w}$, and has length n_w. Phrase 2 is now constructed using the new window and data $X_{n_w+L_1+1}, X_{n_w+L_1+2}, \cdots$. The process is repeated until the N symbols are exhausted. The phrases are $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \cdots, \boldsymbol{Y}_c$ and their lengths are $L_1, L_2, \cdots, L_c$, respectively. The last phrase $\boldsymbol{Y}_c$ is terminated prematurely in the obvious way when the data are exhausted.

From (3), the total number of bits to encode $\boldsymbol{X}_1^N$ is

$$\begin{aligned} \nu(\boldsymbol{X}_1^N) = &\sum_{i=1}^{c} \{|e(L_i)| + \min(\log n_w, \lceil L_i \log A \rceil)\} \\ &+ \lceil n_w \log A \rceil \end{aligned} \tag{4}$$

where the last term is overhead for the first window. Note that c and the $\{L_i\}_{i=1}^c$ are random variables. The average rate is $\overline{R}(N) \triangleq E[\nu(\boldsymbol{X}_1^N)/N]$. Using the bound of (3) we have

$$\boxed{\begin{aligned} \overline{R}(N) \leq &\frac{\lceil n_w \log A \rceil}{N} \\ &+ \frac{1}{N} E \sum_{i=1}^{c} \min(\log n_w + \gamma_1 \log(L_i + 1), \gamma_2 L_i) \end{aligned}}. \tag{5}$$

We are now ready to state our main result.

Theorem 2.1 $\lim_{n_w \to \infty} \lim_{N \to \infty} \overline{R}(N) = H$

It is interesting to note that the version of the sliding-window LZ algorithm used by Stac in their Stacker program

[3] does not encode the length L_i of the ith phrase using $O(\log L_i)$ bits. In fact, they encode L_i using

$$4 + 4\lceil \frac{L_i - 7}{15} \rceil \sim \frac{4L_i}{15}$$

as $L_i \to \infty$. With this encoding, Theorem 2.1 will not hold. We should point out that their commercial algorithm was designed and optimized for a particular window size n_w.

The proof of Theorem 2.1 is given in the remainder of this paper. Some of the ideas were used in a previous paper by the authors [5] on the asymptotic optimality of a fixed database version of LZ.

III. Mathematical Preliminaries

In this section we will derive a result of Wyner and Ziv [4] which appeared in 1989. The special result that we will need here is easily obtained from well-known facts so that we will give a complete proof.

To begin with, we will state the well-known AEP (see, for example, [1, section 15.7]). Let $\{X_k\}_{-\infty}^{\infty}$ be our stationary ergodic random process which takes values in the finite set $\mathcal{A}$. For $\boldsymbol{x} \in \mathcal{A}^\ell$, let

$$P_{\boldsymbol{X}}(\boldsymbol{x}) = P_{\boldsymbol{X}}^\ell(\boldsymbol{x}) = \Pr\{\boldsymbol{X}_1^\ell = \boldsymbol{x}\}.$$

For $\ell = 1, 2 \cdots$, and $\delta > 0$, let

$$T_\ell(\delta) = \left\{ \boldsymbol{x} : \left| \frac{1}{\ell} \log P_{\boldsymbol{X}}^\ell(\boldsymbol{x}) - H \right| < \delta \right\} \tag{6}$$

where H is the entropy of $\{X_n\}$. Then the AEP is

Theorem 3.1 For fixed $\delta > 0$

$$Pr\{X_1^\ell \in T_\ell(\delta)\} \to 1, \quad \text{as } \ell \to \infty.$$

The next result that we will need is due to Kac (see [4, Appendix]). Let $\{U_n\}_{n=-\infty}^{\infty}$ be a stationary ergodic process where the U_n take values in the set $\mathcal{U}$. For $b \in \mathcal{U}$ such that $\Pr\{U_0 = b\} > 0$, define the probability distribution

$$\begin{aligned} Q_b(i) = \ & \Pr\{U_{-i} = b, U_j \neq b, \\ & \text{for } -i < j \leq -1 | U_0 = b\}, \ i = 1, 2, 3, \cdots. \end{aligned} \tag{7}$$

In other words, $Q_b(i)$ is the conditional probability (given $U_0 = b$) that the *first* recurrence of b (in the "backwards" direction) is at $-i$. Since $\{U_n\}$ is ergodic,

$$\sum_{i=1}^{\infty} Q_b(i) = 1.$$

We now establish

Theorem 3.2 (Kac) For $b \in \mathcal{U}$, $\Pr\{U_0 = b\} > 0$

$$\sum_{i=1}^{\infty} i Q_b(i) = 1/\Pr\{U_0 = b\}.$$

Proof: Define the event

$$\begin{aligned} A_{jk} = \{ & U_{-j} = b, U_{+k} = b, U_i \neq b, \\ & \text{for all } i \in (-j, k)\} \end{aligned} \tag{8}$$

for $j = 1, 2, \cdots, k = 0, 1, 2, \cdots$. This is the event that the smallest $n \geq 0$ for which $U_n = b$ is $n = k$, and the largest $n < 0$ for which $U_n = b$ is $n = -j$. The $\{A_{jk}\}$ are disjoint and $\Pr \cup_{j,k} A_{jk} = 1$, by ergodicity. Thus

$$\begin{aligned} 1 &= \Pr \bigcup_{j,k} A_{jk} = \sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \Pr\{A_{jk}\} \\ &= \sum_{j=1}^{\infty} \sum_{k=0}^{\infty} \Pr\{U_k = b\} \ \Pr\{U_{-j} = b, U_i \neq b, \\ & \qquad\qquad \text{for } -j < i < k \mid U_k = b\} \\ & \overset{(a)}{=})\Pr\{U_0 = b\} \sum_{j=1}^{\infty} \sum_{k=0}^{\infty} Q_b(j + k) \\ & \overset{(b)}{=} \Pr\{U_0 = b\} \sum_{i=1}^{\infty} i Q_b(i). \end{aligned} \tag{9}$$

Step (a) follows from the stationary of $\{U_n\}$, and step (b) on observing that $Q_b(i)$ appears in the double sum exactly i times for $(j, k) = (1, i-1), (2, i-2), \cdots, (i, 0)$. Equation (9) is Theorem 3.2.

Now for an infinite sequence $\boldsymbol{x} = \boldsymbol{x}_{-\infty}^{\infty}$, where $x_n \in \mathcal{A}$, define for $\ell > 0$

$$W_\ell(\boldsymbol{x}) = \text{ smallest } k > 0 \text{ such that } \boldsymbol{x}_0^{\ell-1} = \boldsymbol{x}_{-k-\ell+1}^{-k}. \tag{10}$$

Thus $W_\ell(\boldsymbol{x})$ is the number of positions that we have to "slide" $\boldsymbol{x}_0^{\ell-1}$ to the left to get a perfect match. For example, with

$$\begin{array}{llllll|lllll} n: & -5 & -4 & -3 & -2 & -1 & 0 & 1 & 2 & 3 & 4 \\ x: \cdots & a & b & c & a & b & c & a & b & c & d \ \cdots \end{array}$$

$W_4(\boldsymbol{x}) = 3$, since $\boldsymbol{x}_{-3}^{0} = \boldsymbol{x}_0^3$. Now with $\{X_n\}$ as defined in Section II, define for $-\infty < n < \infty$, the ℓ-tuple

$$U_n = (X_n, X_{n+1}, \cdots, X_{n+\ell-1}) \tag{11}$$

where $\ell > 0$ is a fixed integer. The sequence $\{U_n\}$ is also stationary and ergodic. Furthermore

$$\Pr\{W_\ell(\boldsymbol{X}) = i \mid \boldsymbol{X}_0^{\ell-1} = z\} = Q_{\boldsymbol{z}}(i) \tag{12}$$

$1 \leq i < \infty$, where $Q_{\boldsymbol{z}}$ is defined by (7). Thus for $n = 1, 2, \cdots$, by the Markov inequality and Theorem 3.2

$$\begin{aligned} \Pr\{W_\ell > n \mid \boldsymbol{X}_0^{\ell-0} = \boldsymbol{z}\} &\leq \frac{E(W_\ell | \boldsymbol{X}_0^{\ell-1} = z)}{n} \\ &= \frac{1}{n \Pr\{\boldsymbol{X}_0^{\ell-1} = z\}}. \end{aligned} \tag{13}$$

We are now ready to give the final theorem of this section.

Theorem 3.3 For $\ell = 1, 2, \cdots,$ *and* $\epsilon > 0$ *arbitrary*

$$Pr\{W_\ell(\boldsymbol{X}) > 2^{\ell(H+\epsilon)}\} \to 0, \quad as\ \ell \to \infty$$

where H *is the entropy of* $\{X_n\}$.

Proof: Write, for fixed $\epsilon > 0, \delta = \epsilon/2$, and $T_\ell(\delta)$ defined by (6)

$$\begin{aligned}
&\Pr\{W_\ell > 2^{\ell(H+\epsilon)}\} \\
&= \sum_{\boldsymbol{z}\in T_\ell(\delta)} \Pr\left\{\boldsymbol{X}_0^{\ell-0} = \boldsymbol{z}\right\} \Pr\left\{W_\ell > 2^{\ell(H+\epsilon)} | \boldsymbol{X}_0^{\ell-1} = \boldsymbol{z}\right\} \\
&\quad + \sum_{\boldsymbol{z}\neq T_\ell(\delta)} \Pr\{\boldsymbol{X}_0^{\ell-1} = \boldsymbol{z}\} \Pr\{W_\ell > 2^{\ell(H+\epsilon)} | \boldsymbol{X}_0^{k-\ell} = \boldsymbol{z}\} \\
&\quad \leq \sum_{z\in T_\ell(\delta)} \Pr\{\boldsymbol{X}_0^{\ell-1} = \boldsymbol{z}\} \frac{1}{2^{\ell(H+\epsilon)} \Pr\{\boldsymbol{X}_0^{\ell-1} = \boldsymbol{z}\}} \\
&\quad + \Pr\{\boldsymbol{X}_0^{\ell-1} \notin T_\ell(\delta)\}. \qquad (14)
\end{aligned}$$

The inequality in (14) follows from (13) with $n = 2^{\ell(H+\epsilon)}$. Now for $\boldsymbol{z} \in T_\ell(\delta)$, $\Pr\{\boldsymbol{X}_0^{\ell-1} = \boldsymbol{z}\} \geq 2^{-\ell(H+\delta)}$. Thus from (14), with $\epsilon - \delta = \epsilon/2$

$$\begin{aligned}
\Pr\{W_\ell > 2^{\ell(H+\epsilon)}\} &\leq 2^{-\ell(\epsilon-\delta)} \Pr\{\boldsymbol{X}_0^{\ell-1} \in T_\ell(\delta)\} \\
&\quad + \Pr\{\boldsymbol{X}_0^{\ell-1} \notin T_\ell(\delta)\} \to 0, \\
&\qquad \text{as } \ell \to \infty \qquad (15)
\end{aligned}$$

by Theorem 3.1 (AEP). This is Theorem 3.3.

We will use the following form of Theorem 3.3:

Corollary 3.4 Let $n(\ell)$, $\ell = 1, 2, \cdots$, satisfy

$$L \triangleq lim_{\ell\to\infty} \frac{1}{\ell} log\, n(\ell) > H. \qquad (16a)$$

Then

$$lim_{\ell-\infty} Pr\{W_\ell > n(\ell)\} = 0. \qquad (16b)$$

Proof: Let $\epsilon = L - H > 0$. For $\ell \geq \ell_0$ (sufficiently large), $(1/\ell)\log n(\ell) \geq L - (\epsilon/2) = H + (\epsilon/2)$. Thus

$$n(\ell) \geq 2^{\ell[H+(\epsilon/2)]}.$$

Thus for $\ell \geq \ell_0$

$$\Pr\{W_\ell > n(\ell)\} \leq \Pr\left\{W_\ell > 2^{\ell[H+(\epsilon/2)]}\right\} \to 0$$

by Theorem 3.3.

IV. Proof of Theorem 2.1

That $\overline{R}(N) \geq H$ for all N follows from the converse to the Coding Theorem for lossless source coding. Since the algorithm defined in Section II yields a prefix-free ("instantaneous") code for $X_1, \cdots, X_N$

$$E\nu(\boldsymbol{X}_1^N) \geq H(X_1, \cdots, X_N).$$

(See Theorem 5.3.1 in [1].) Since $H(X_1, \cdots, X_N) \geq NH, \bar{R}(N) = E\nu(\boldsymbol{X}_1^N)/N \geq H$.

We now show that lim $\overline{R}(N) \leq H$. Suppose that we have implemented the algorithm to encode $X_1, \cdots, X_N$. Subdivide the interval $[n_w + 1, N]$ into N'/ℓ_0 subintervals of length ℓ_0, where

$$N' = N - n_w \qquad (17a)$$

and

$$\ell_0 = \frac{\log n_w}{H + \epsilon} \qquad (17b)$$

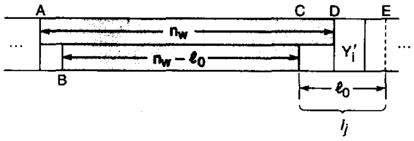


Fig. 1.

and $\epsilon > 0$ is arbitrary. (Assume that ϵ is adjusted so that ℓ_0 is an integer, and that ℓ_0 divides N'). Recall that $\boldsymbol{X}_1^{n_w}$ is the (uncoded) initial window. Denote the N'/ℓ_0 subintervals by $\{I_j\}$, where

$$I_j = [n_w + (j-1)\ell_0 + 1, n_w + j\ell_0], \quad j = 1, 2, \cdots, N'/\ell_0.$$

We say that a subinterval I_j is *bad* if a copy of $\{X_k\}_{k\in I_j}$ does not begin in the string of $(n_w - \ell_0)$ symbols preceding it, i.e.

$$\boldsymbol{X}_{n_w+(j-1)\ell_0+1}^{n_w+j\ell_0} \neq \boldsymbol{X}_{n_w+(j-1)\ell_0+1-m}^{n_w+j\ell_0-m}, \quad 1 \leq m \leq n_w - \ell_0. \qquad (18)$$

Now referring to the discussion in Section III, we have that

$$\Pr\{I_j \text{ is bad}\} = \Pr\{W(\ell_0) > n_w - \ell_0\}. \qquad (19)$$

Since $n(\ell_0) \triangleq n_w - \ell_0 = 2^{\ell_0(H+\epsilon)} - \ell_0$ satisfies (16a), with $L = H + \epsilon$, Corollary 3.4 implies that

$$\Pr\{I_j \text{ is bad }\} \to 0, \qquad (20)$$

as $n_w($ and $\ell_0) \to \infty$.

Now let $\{\boldsymbol{Y}_i\}, 1 \leq i \leq c$ be the phrases generated by the algorithm when encoding $\boldsymbol{X}_1^N$. For $1 \leq i \leq c$, let $\boldsymbol{Y}'_i$ be the phrase $\boldsymbol{Y}_i$ augmented by the next symbol from $\boldsymbol{X}_1^N$. Thus if $Y_i = \boldsymbol{X}_k^{k+L_i-1}$, then $\boldsymbol{Y}'_i = \boldsymbol{X}_k^{k+L_i}$. We say that the phrase $\boldsymbol{Y}_i$ is in *internal* phrase, if the corresponding augmented phrase $\boldsymbol{Y}'_i$ begins and ends in the same subinterval I_j. We claim that if $\boldsymbol{Y}_i$ is an internal phrase supported on I_j, then I_j is a bad subinternal. We show this with the aid of the schematic diagram in Fig. 1. The figure shows the augmented internal phrase $\boldsymbol{Y}'_i$ supported on the interval I_j. The window corresponding to phrase $\boldsymbol{Y}_i$ is supported on the interval AD. By definition, a copy of $\boldsymbol{Y}'_i$ does *not* begin in this window (since $\boldsymbol{Y}_i$, is the *longest* sequence, beginning at D, a copy of which begins in window). Since $\boldsymbol{Y}'_i$ is supported on I_j, a copy of $\{X_k\}_{k\in I_j}$ cannot begin in the window AC, and therefore cannot begin in BC (since B is to the right of A). Thus I_j is a bad subinterval.

Let $S_I = \{i : \boldsymbol{Y}_i \text{ is internal}\}$, and let ψ be the fraction of the intervals $\{I_j\}$ which are bad. It follows that

$$\begin{aligned}
\sum_{i\in S_I} |Y_i| &\leq \ell_0 \{\text{no. of bad intervals}\} \\
&\leq \ell_0 \frac{N'}{\ell_0} \psi = N'\psi. \qquad (21)
\end{aligned}$$

We are now ready to complete the proof to Theorem 2.1. Rewrite inequality (5) as

$$\overline{R}(N) \le \frac{\lceil n_w \log A \rceil}{N} + \frac{1}{N}E\left\{ \sum_{i\in S_I} \gamma_2 L_i + \sum_{i \in S_I^c} [\log n_w + \gamma_1 \log (L_i + 1)] \right\} \tag{22}$$

Now the first expectation is from (21)

$$\frac{1}{N}E\sum_{i\in S_I} \gamma_2 L_i \le \frac{\gamma_2 N'}{N} E\psi \le \gamma_2 \Pr\{I_j \text{ is bad}\}. \tag{23}$$

Finally, observe that if $\boldsymbol{Y}_i$ is *not* an internal phrase, then $\boldsymbol{Y}_i$ must occupy the last position of some subinterval I_j. Thus

$$c' \triangleq |S_I^c| \le \{\text{number of subintervals}\} = \frac{N'}{\ell_0}. \tag{24}$$

The second term in (22) (without the expectation) is

$$\begin{aligned} &\frac{1}{N}\sum_{i\in S_I^c} \log n_w + \gamma_1 \frac{1}{N}\sum_{i\in S_I^c} \log (L_i+1) \\ &= |S_I^c| \frac{\log n_w}{N} + \gamma_1 \frac{c'}{N} \sum_{i \in S_I^c} \frac{1}{c'} \log (L_i + 1) \\ &\overset{(a)}{\le} |S_I^c| \frac{\log n_w}{N} + \gamma_1 \frac{c'}{N} \log\left(\frac{1}{c'}\sum L_i + 1\right) \\ &\overset{(b)}{\le} \frac{N'}{N}\frac{\log n_w}{\ell_0} + \gamma_1 \frac{c'}{N}\log\left(\frac{N}{c'}+1\right) \\ &\overset{(c)}{\le} \frac{\log n_w}{\ell_0} + \frac{\gamma_1}{\ell_0}\log(\ell_0+1) \\ &\overset{(d)}{=} (H+\epsilon) + (\gamma_1/\ell_0)\log(\ell_0+1). \end{aligned} \tag{25}$$

Step (a) in (25) follows from the concavity of $\log(\cdot)$; step (b) from $\sum_{i\in S_I^c} L_i \le N$; step (c) from the fact that $(1/x)\log(x+1), x \ge 0$, is decreasing in x, and from (24) which implies

$$\frac{c'}{N} \le \frac{N'}{N\ell_0} \le \frac{1}{\ell_0}$$

and step (d) from (17b).

Substituting (23) and (25) into (22) we have

$$\overline{R}(N) \le \frac{\lceil n_w \log A\rceil}{N} + \gamma_2 \ \Pr\{I_j \text{ bad }\} + H + \epsilon + (\gamma_1/\ell_0)\log(\ell_0+1)$$

and

$$\lim_{N\to\infty} \overline{R}(N) \le \gamma_2 \ \Pr\{I_j \text{ is bad}\} + H + \epsilon + (\gamma_1/\ell_0)\log(\ell_0+1).$$

Finally, letting $n_w \to \infty$, and using (20) we have

$$\lim_{n_w\to\infty}\lim_{N\to\infty} \overline{R}(N) \le H + \epsilon$$

which is Theorem 2.1, on letting $\epsilon \to 0$.

Appendix
Comma-Free Coding of an Unbounded Integer

In this Appendix we define a scheme for unambiguously encoding an integer L into a binary string. Let $\{0,1\}*$ be the set of binary sequences of arbitrary length. We will give an encoding or mapping $e : [1,\infty) \to \{0,1\}*$, which maps the nonnegative integers into binary sequences, such that for any distinct $L_1, L_2, e(L_1)$ is not a prefix of $e(L_2)$. Thus the code is uniquely decipherable, see for example [1, ch. 5].

For $k \in [1,\infty)$ let $b(k)$ be the binary expansion of the integer k. Thus $b(1) = 1, b(2) = 10$, etc. The length of this expansion is

$$|b(k)| = \lceil \log(k+1) \rceil. \tag{A1}$$

Also, for $k = 1, 2, \cdots$, let $u(k) \triangleq 0^{k-1}1$, the concatenation of $(k-1)$ 0's followed by a 1. First observe that

$$\hat{e}(k) = u(|b(k)|) * b(k) \tag{A2}$$

where * denotes concatenation, is a prefix-free encoding of $k \in [1,\infty)$. For example, if $\hat{e}(k_1) = 00011001$, the prefix 0001 tells us that the next 4 bits, 1001, are the binary encoding of $k_1 = 9$.

The mapping which we will use is, for $L = 1, 2, \cdots$

$$e(L) = \hat{e}(|b(L)|) * b(L). \tag{A3}$$

The prefix $\hat{e}(|b(L)|)$ encodes the length of $b(L)$, and the next $|b(L)|$ bits is the binary expansion of L. For example, when $L = 7, b(L) = 111$, and $|b(L)| = 3$ so that $\hat{e}(|b(L)|) = 0111$. Thus $e(L) = e(7) = 0111111$.

We need a bound on $|e(L)|$. From (A2) $|\hat{e}(k)| = 2|b(k)|$. Thus from (A3),

$$|e(L)| = 2|b(|b(L)|)| + |b(L)|.$$

Using (A1) we see that for large L,

$$|e(L)| \sim \log L + 2\log\log L.$$

We can account for small values of L by writing, for *all* $L = 1, 2, \cdots$,

$$|e(L)| \le \log(L+1) + \gamma \log\log(L+2) \tag{A4a}$$

for a suitably large γ. For our purposes the very weak bound

$$|e(L)| \le \gamma \log(L+1) \tag{A4b}$$

will suffice, where γ is appropriately large, say $\gamma = 10$.

References

[1] T. Cover and J. Thomas, *Elements of Information Theory.* New York: Wiley, 1991.

[2] R. Gallager, *Information Theory of Reliable Communication.* New York: Wiley, 1968.

[3] D. Whiting, G. George, and G. Ivey, "Data compression apparatus and method," US Patent 5016009, 1991 (assigned to Stac, Inc.).

[4] A. D. Wyner and J. Ziv, "Some asymptotic properties of the entropy of a stationary ergodic data source with applications to data compression," *IEEE Trans. Informat. Theory,* vol. 36, pp. 1250–1258, Nov. 1989.

[5] ——, "Fixed data base version of the Lempel–Ziv data compression algorithm," *IEEE Trans. Informat. Theory*, vol. 37, pp. 878–880, May 1991.

[6] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Informat. Theory*, vol. IT-24, pp. 337–343, May 1977.

Aaron D. Wyner (Fellow, IEEE) received the B.S. degree in 1960 from Queens College, City University of New York, and the B.S.E.E., M.S., and Ph.D. degrees from Columbia University, New York, in 1960, 1961, and 1963, respectively.

Since 1963 he has been a member of the Mathematical Sciences Research Center at AT&T Bell Laboratories, Murray Hill, NJ, doing research in information and communication theory and related mathematical areas. From 1974 to 1993 he was Head of the Communications Analysis Research Department at Bell Labs. He spent the year 1969–1970 visiting the Department of Applied Mathematics at the Wietzmann Institute of Technology, Haifa, Israel, and the Faculty of Electrical Engineering at the Technion–Israel Institute of Technology, also in Haifa, on a Guggenheim Foundation Fellowship. He has also been a full- and part-time faculty member at Columbia University, Princeton University, and the Polytechnic Institute of Brooklyn. In 1977 he received the IEEE Information Theory Society Prize Paper award and in 1984 the IEEE Centennial Medal.

Dr. Wyner was Chairman of the IEEE Information Theory Society's Metropolitan New York Chapter, Editor-in-Chief of the *IEEE Transactions on Information Theory*, and Cochairman of two international symposia. In 1976 he served as President of the IEEE Information Theory Society. He is a member of URSI, and served as Vice-Chair of the International Commission C, and also is a member of the National Academy of Engineering.

Jacob Ziv (Fellow, IEEE) was born in Tiberias, Israel, on November 27, 1931. He received the B.Sc., Dipl. Eng., and M.Sc. degrees, in electrical engineering, from Technion–Israel Institute of Technology, Haifa, in 1954 and 1957, respectively, and the D.Sc. degree from the Massachusetts Institute of Technology, Cambridge, in 1962.

From 1955 to 1959 he was a Senior Research Engineer in the Scientific Department of the Israel Ministry of Defense, and was assigned to the research and development of communication systems. From 1961 to 1962, while studying for the Ph.D. degree at MIT, he joined the Applied Science Division of Melpar, Inc., Watertown, MA, where he was a Senior Research Engineer doing research in communication theory. In 1962 he returned to the Scientific Department, Israel Ministry of Defense, as Head of the Communications Division, and was also an Adjunct at the Faculty of Electrical Engineering at the Technion. From 1968 to 1970 he was a Member of the Technical Staff at Bell Laboratories, Murray Hill, NJ. He joined the Technion in 1970 and is Herman Gross Professor of Electrical Engineering there. He was Dean of the Faculty of Electrical Engineering from 1974 to 1976 and Vice President for Academic Affairs from 1978 to 1982. From 1977 to 1978, 1983 to 1984, and 1991 to 1992 he was on sabbatical leave at Bell Laboratories. In 1982 he was elected Member of the Israeli Academy of Science, and was appointed as a Technion Distinguished Professor. In 1993 he was awarded the Israel Prize in Exact Sciences (Engineering and Technology). He has twice been the recipient of the IEEE-Information Theory Best Paper Award (for 1976 and 1979). He was the Chairman of the Israeli Universities Planning and Grants Committee from 1985 to 1991. His research interests include general topics in data compression, information theory, and statistical communication.

In 1988, Dr. Ziv was elected as a Foreign Associate to the U.S. National Academy of Engineering.