A New Universal Coding Scheme for the Binary Memoryless Source

JOHN C. LAWRENCE

Abstract—A coding scheme for the binary memoryless source is presented. The scheme is an extension of a scheme discovered by Lynch and Davisson and developed by Schalkwijk, and is based upon the Schalkwijk algorithm for the Lynch–Davisson (LD) code. The new coding scheme is shown to be asymptotically optimal as the block length approaches infinity, regardless of the source statistics. Although the LD scheme is minimax-universal, it is not optimal for low source entropies and finite block lengths. Run length coding is shown to be closer to optimal for low source entropies, and the new scheme is shown to be practically equivalent to run length coding in this range.

I. Introduction

A SOURCE CODING scheme was proposed by Lynch [1] and Davisson [2] in 1966. In 1972, Schalkwijk [3] presented an algorithm for coding binary sources. Davisson [4] pointed out that the Schalkwijk scheme was similar to the Lynch-Davisson (LD) scheme. The basic algorithm used was the same in each case. However, Lynch and Davisson implemented the algorithm as a block-to-variable scheme which does not require knowledge of the source statistics. The variable-to-block Schalkwijk implementation does require knowledge of the source statistics. Schalkwijk's implementation is based on the ranking of fixed-weight binary sequences, whereas the LD scheme is based on the ranking of fixed-length sequences.

Schalkwijk also indicated how the algorithm could be extended to nonbinary sources. The original LD algorithm was for generalized sources, but it was basically a binary algorithm with a capability for handling nonbinary sources tacked on.

In his conclusions, Schalkwijk pointed out that with a slight modification, his algorithm could be used for block-to-variable length coding, although he did not develop this case. It is this form of the Schalkwijk algorithm that is equivalent to the binary case of the LD algorithm.

Later, Davisson [5] showed that the block-to-variable algorithm is asymptotically optimal for the binary memoryless source. He also showed that the scheme is minimax-universal, the strongest and most desirable kind of universal coding.

In this paper, we start with the block-to-variable algorithm and point out its limitation for finite block lengths. The discussion is slanted towards Schalkwijk's approach, since his development is considered to be pedagogically more transparent. A new coding scheme based on the same underlying algorithm is presented. This scheme overcomes the limitation of the LD scheme for finite block lengths. The performance of our scheme is compared with that of the LD scheme and also with run length coding.

Manuscript received May 27, 1975; revised November 15, 1976.
The author is at 1815 Gatepost Rd., Encinitas, CA 92024.

II. Schalkwijk Algorithm for the LD Code

In the exposition of the Schalkwijk algorithm for the LD code, we let the incoming sequence of source bits determine a random walk in Pascal's triangle starting at the apex and proceeding down a number of rows equal to the block length. Details of the algorithm are given in Appendix A. The encoding process can terminate on any point of the last row.

The decoder must know which point the encoding process terminated on, since this is the "starting point" for the decoding process. Therefore, to distinguish among possible starting points, a fixed-length prefix is provided. The binary representation of the running sum, which is computed in the encoding process, becomes the suffix. If the input block length is N bits, the length of the prefix will be equal to the greatest integer which has a value less than $\log_2 (N + 1) + 1$ bits. The number of bits required for the suffix is equal to the greatest integer which has a value less than $\log_2 \binom{N}{w} + 1$ bits, where w is the weight of the block. The length of the suffix will vary depending on the starting point.

Davisson's results [5, pp 786–787] which were derived for "sequence time coding," [1], [2] are also applicable to the Schalkwijk block-to-variable scheme, since the schemes are identical. It is assumed that the binary memoryless source is characterized by the fixed but unknown parameter p which is equal to the probability of a one. Following Davisson, the per letter coding rate is at most

$$\frac{1}{N}(\log_2 (N + 1) + 2) + \frac{1}{N}\log_2 \binom{N}{w}, \qquad 0 \le w \le N. \quad (1)$$

The first term goes to zero as $N \to \infty$. The second term converges to $H(p) = -p \log_2 p - (1 - p) \log_2 (1 - p)$, where $p = \lim_{N\to\infty} (w/N)$. Therefore, the scheme is asymptotically optimum in terms of Shannon's noiseless coding theorem [6]. The scheme is also minimax-universal, which is Davisson's term for the strongest and most de-

sirable kind of universal coding. When a coding scheme is minimax-universal, the user can transmit the source output at a rate per symbol arbitrarily close to the source entropy rate, uniformly over all values of p, by taking N large enough. The proof that the block-to-variable coding scheme is minimax-universal is presented in Davisson's paper [5, p. 789].

III. Limitation of the LD Coding Scheme

The main limitation of the LD coding scheme is its performance at low entropies for finite block lengths. Consider the per letter rate as $p \to 0$. The second term in (1) approaches zero, while the first term remains constant.

The presence of the first term in (1) is due to the prefix in the coding scheme. This term limits the minimum per letter coding rate or, by the same token, the maximum compression that can be achieved at low entropies.

Let us compare a simple ad hoc scheme, such as run length coding, with the LD scheme. For the run length scheme, a block of N bits contains the binary representation of the length of the run of zeroes.

We take the relative measure of complexity for all schemes considered in this paper to be the block length. Therefore, when making a comparison, we will always consider schemes of the same block length.

For example, let us assume that $N = 37$. Also, let us assume an incoming bit sequence consisting of 740 zeroes. The run length scheme will encode the 740 zeroes in one block, and the resultant compression ratio is $740/37 = 20$. On the other hand, the LD encoder will segment the incoming sequence into 20 subsequences of 37 zeroes each and encode each subsequence separately. A six-bit prefix is required since there are 38 possible weights in a 37-bit block, but no suffix is required. The compression ratio for each block will be $37/6 = 6.17$, which will also be equal to the overall compression ratio.

As the preceding example makes clear, the block length of the LD scheme, and hence the complexity, would have to be increased considerably before the performance would compare favorably with run length coding for low entropy sequences.

IV. Maximum Entropy Variable-to-Block Coding

The coding scheme presented herein overcomes the limitation of the LD block-to-variable scheme. It is called maximum entropy variable-to-block coding, a term which will be explained later. First, an explanation of the scheme is given. Then, it is proved that the scheme is asymptotically optimal. Finally, we show that the low entropy performance is practically as good as that of run length coding.

Again, we visualize the coding process as a random walk in Pascal's triangle, starting at the apex and proceeding downward until a boundary is reached. When the boundary is reached, the input is terminated and the running sum, which has been computed according to the algorithm, becomes the suffix of the output block. Let S be equal to the suffix length.

The essential difference between this scheme and the Schalkwijk scheme is the way in which the boundary is defined. Let (n,w) denote the wth element of the nth row of Pascal's triangle. The boundary consists of the elements (n,w^*) and is defined such that the following conditions hold

$$\log_2 \binom{n}{w^*} \le S,$$

$$\log_2 \binom{n+1}{w^*+1} > S, \qquad \text{for } w \le \frac{n}{2},$$

$$\log_2 \binom{n+1}{w^*} > S, \qquad \text{for } w > \frac{n}{2}. \tag{2}$$

With reference to Fig. 1, the preceding conditions can be intuitively interpreted as follows. Assume that the random walk has proceeded until the element (n,w) has been reached and that $\binom{n}{w} \le 2^S$. Also, we assume that $w < n/2$, i.e., the run contains more zeroes than ones. Then, if the next element is a one, the random walk would proceed one step in the $-Y$ direction to the element $(n+1, w+1)$. If the value of this element $\binom{n+1}{w+1}$ exceeds 2^S, then the element (n,w) is a boundary element. If the encoding procedure terminates on a boundary element, then the encoded word can always be expressed in S bits, since there are at most 2^S possible ways of reaching a boundary element. Fig. 1 illustrates the construction of the boundary in Pascal's triangle for $S = 31$. Boundary elements are denoted z. Nonboundary elements are denoted by their numerical value or the letter x. Elements exceeding 2^{31} are denoted by "0."

The first 32 rows contain no boundary elements. The first boundary elements occur in the 33rd row: (33,15), (33,16), (33,17), and (33,18). Row 34 contains three boundary elements for which $w \le 17$ and three for which $w > 17$. Row 35 contains four boundary elements; row 36, two boundary elements; etc. For rows greater than the 36th, there will be either two or four boundary elements per row. The last boundary elements are the first and last two elements of the 2^{31} st row, i.e., $(2^{31},1)$ and $(2^{31},2^{31}+1)$.

We divide the boundary elements into sets as follows. Consider two boundary elements (n_a,w_a^*) and (n_b,w_b^*). If $w_a^* = w_b^*$, then the boundary elements belong to the same set. Let n_1 be the row number of the row containing the first boundary elements. Then, there will be a total of $n_1 + 1$ sets of boundary elements. In the example of Fig. 1, there are 34 sets.

A significant difference between the LD scheme and the scheme presented herein is the way in which runs are allowed to terminate. In the Schalkwijk algorithm for the LD

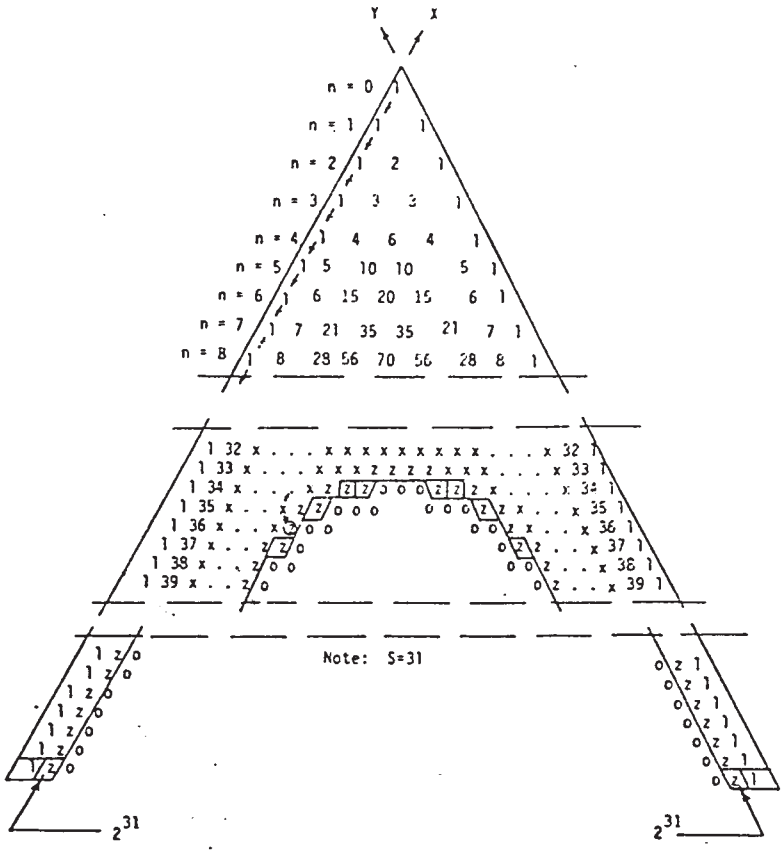

Fig. 1. Construction of boundary in Pascal's triangle.

scheme, a source run could terminate on any boundary element, i.e., every boundary element is a "starting point." In our scheme, only one starting point is allowed for each set of boundary elements. How the source runs are made to terminate only on starting points will be discussed later. The starting point for each set of boundary elements is defined to be that boundary element (n^*,w^*) such that $n^* \geq n$ for all boundary elements (n,w^*) in that set. All starting points are shown in boxes in Fig. 1.

Consider the set of binary sequences of length n and weight w. The set has $\binom{n}{w}$ elements. Now consider the set of such sets where n takes on a different value for each set, $w \leq n < \infty$, and w is fixed. The set for which n and w correspond to a starting point contains the largest number of members, each of which can still be represented in S bits. This follows from the facts that $\log_2 \binom{n^*}{w^*} \leq S$ and $\log \binom{n^*+1}{w^*} > S$. For this reason, we call our scheme the "maximum entropy" variable-to-block scheme.

The total number of starting points is related to the suffix length S in the following way. We have the inequality $\binom{S}{S/2} \leq 2^S < \binom{2S}{S}$ for S even. For S odd, we have $\binom{S}{[S+1]/2} \leq 2^S < \binom{2S}{S}$. From this it follows that, if n_1 is the number of the row in which the first boundary element occurs, $S \leq n_1 + 1 \leq 2S$.

Note that the shortest encoded source sequence will be at least $S-1$ bits in length. The total number of sets of boundary elements (which is equal to $n_1 + 1$) is at most $2S$. Since there is one starting point per set, the total number of starting points is at most $2S$ and, therefore, can be specified in a prefix of at most $\log_2 S + 2$ bits. Since the construction of the starting points in Pascal's triangle is perfectly symmetrical, we can set aside one bit of the prefix to specify whether the random walk dictated by the incoming source bits reaches the boundary in the left or right half. This 1-bit prefix then will specify a "0-coded" or "1-coded" run. The starting points on the left side are numbered consecutively from top to bottom and their rank determines the remaining bits of the prefix. The entire encoded block consists of at most $N = S + \log_2 S + 2$ bits.

The computation of the suffix is exactly the same as in the Schalkwijk algorithm for the LD scheme. Starting at the apex, the random walk dictated by the incoming source bits proceeds downward in Pascal's triangle, moving a step in the $-X$ direction when the incoming bit is a zero and a step in the $-Y$ direction when the incoming bit is a one. Additionally, if the incoming bit is a one, the value of the

element one step in the X direction from the element which has been moved to is added to the running sum which equals the value of the suffix when the process terminates.

To simplify the discussion, we will assume that the run contains more zeroes than ones so that we can confine our attention to the left half of Pascal's triangle. Eventually, the random walk will reach a boundary element. At this point, no more source bits can be accepted. Since, in general, we are not yet at a starting point, we add "dummy" [1] zeroes to the source sequence, advancing one row in Pascal's triangle with the addition of each zero until a starting point is reached. This addition of "dummy" zeroes insures that every sequence which is coded terminates on a starting point. At the decoder, the starting point is determined by decoding the prefix and then the suffix is decoded relative to that starting point. Then all initial consecutive zeroes are stripped off, thus eliminating the dummy bits which were added by the encoder. However, the maximum number of zeroes that can be stripped off without destroying data bits is equal to one less than the number of boundary elements in the set corresponding to the starting point. This precaution must be taken since the boundary element contained in the lowest order row of each set can be approached by either a zero or a one. Every other boundary element in the set can be approached only by a one. Again, the decoded bit sequence must be reversed to obtain the original ordering.

As an example, consider a sequence of 12 zeroes, followed by 12 ones, followed by 11 zeroes, followed by a one. This sequence is indicated by arrows in Fig. 1. This would place us on the 13th element of row 36, which is shown encircled in Fig. 1. The input sequence is terminated at this point. One dummy zero is added to bring us to the starting point, which is the element (37,13). The running sum at this point is equal to $\Sigma_{w=1}^{12} \binom{11+w}{w}$. This becomes the value of the suffix. Since the starting point (37,13) is the fifth starting point from the top, the prefix would be 000101.

V. Optimality of Maximum Entropy Coding Scheme

We shall consider a composite binary source in which the probability p of a one is chosen by nature randomly according to the uniform probability law on (0,1) and then fixed for all time. The probability, given p, that any message block of length n contains w ones is

$$p^w(1-p)^{n-w}. \tag{3}$$

If p were known, a per letter codeword length at least equal to the entropy $H(p) = -p \log_2 p - (1-p) \log_2 (1-p)$ would be required to block encode the source. Although p is unknown, the composite source output probabilities are known. In fact, the probability of any given message containing w ones in n outputs is simply

$$\int_0^1 p^w(1-p)^{n-w}\,dp = \frac{1}{n+1}\binom{n}{w}^{-1}. \tag{4}$$

Davisson [2] has shown that the probabilities of (4) provide codes as good asymptotically as the probabilities of (3) with p known exactly. This means that it is possible to code the binary memoryless source in such a way that the resulting data compression is as great when p is not known as it would be if p were known exactly.

In Appendix B, we prove that the per letter coding rate of the maximum entropy scheme approaches the optimal limit as the block length approaches infinity. Therefore, we have the following.

Theorem 1: For the maximum entropy variable-to-block coding scheme, the following relation holds:

$$\lim_{N\to\infty} \frac{N}{n} = H(p), \qquad 0 \le p \le 1. \tag{5}$$

where n = number of input bits, N = number of output bits, and where the limit is obtained with probability one (wp1).

VI. Performance Characteristics

We now compute the average per letter coding rates for the schemes considered in this paper for the same block length and plot the results. Fig. 2 presents the results for 1) run length coding, 2) Lynch–Davisson coding, and 3) maximum entropy coding. Also, $H(p)$ is shown. The curves are symmetrical about $p = \frac{1}{2}$. Therefore, only the results for $0 \le p \le \frac{1}{2}$ are shown. The block length is 37 bits for all cases. The general nature of the results has already been made clear. LD coding is close to optimal in the vicinity of $p = \frac{1}{2}$, but approaches a nonzero asymptote as $p \to 0$ and $p \to 1$. Run length coding outperforms LD coding for small p. The maximum entropy scheme stays close to optimal for p very close to either zero or one, as well as for p in the vicinity of $\frac{1}{2}$.

The formulas from which the results were computed were derived as follows. Let N equal the block length and n be a variable representing either input run length or output codeword length.

We start with run length coding. A one bit prefix is 0 (1) if the first bit of the incoming sequence is a 0 (1) and the run is all zeroes (ones). This bit is provided so that the run length coding scheme corresponds to the other schemes in which an initial bit specifies either the left or right half of Pascal's triangle, i.e., a "0-coded" or "1-coded" run.

For the purposes of the following explanation, we assume that the source run consists of a string of zeroes terminated by a one. A similar explanation applies if the run consists of a string of ones terminated by a zero. The suffix consisting of $N-1$ bits contains the binary representation of a number which is one less than the number of zeroes in the run. The maximum number of bits that can be accepted is 2^{N-1}. $(2^{N-1}-1)$ zeroes followed by a one is coded

[1] See [3] for another example of the use of dummy bits.

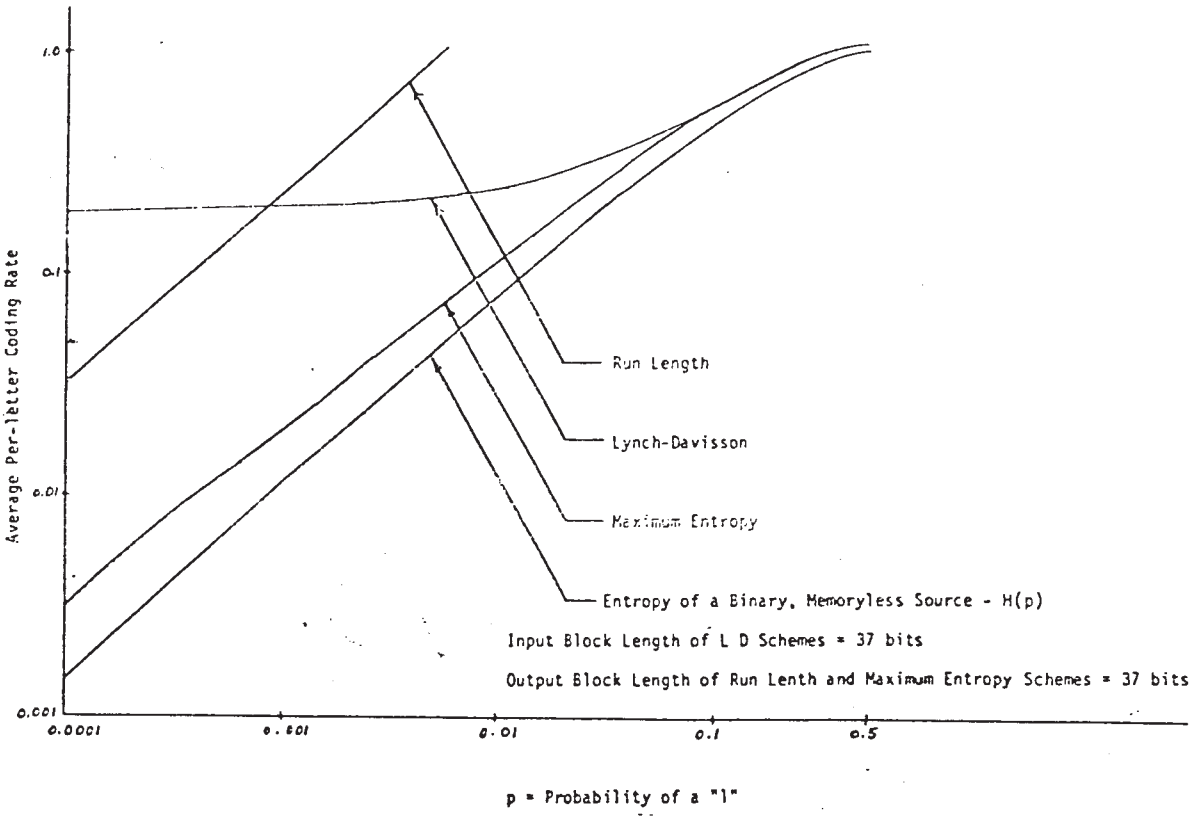

Fig. 2. Average per letter coding rate versus probability p of "1."

as the binary representation of $2^{N-1}-2$, and the all zero run of 2^{N-1} bits is encoded as the binary representation of $2^{N-1}-1$. The expression for average per letter coding rate is

$$\left(\frac{\bar{N}}{n}\right)^R = \sum_{n=2}^{2^{N-1}} \binom{N}{n} \{(1-p)^{n-1}p + p^{n-1}(1-p)\} + \left(\frac{N}{2^{N-1}}\right)\{(1-p)^{2^{N-1}} + p^{2^{N-1}}\}. \quad (6)$$

The minimum run length is two bits (01 or 10). The probability of a run of $n-1$ zeroes and a one is simply $(1-p)^{n-1}p$. There are four possible runs of length 2^{N-1}; one consisting of all zeroes (ones) except the last bit, which is 1 (0) and one consisting of all zeroes (ones). The last term in the above equation accounts for the all zero (one) run.

For the LD scheme, the boundary consists of the elements in the Nth row of Pascal's triangle. The probability of the two-dimensional random walk terminating at the point (N,w) is the probability of w ones in N bits; namely $\binom{N}{w} p^w(1-p)^{N-w}$. The number of encoded bits is equal to $n = \log_2(N+1) + \log_2\binom{N}{w} + 2$. Thus,

$$\left(\frac{\bar{n}}{N}\right)^{\mathrm{LD}} = \frac{1}{N}\sum_{w=0}^{N}\left\{\log_2(N+1) + \log_2\binom{N}{w} + 2\right\} \cdot \left\{\binom{N}{w} p^w(1-p)^{N-w}\right\}. \quad (7)$$

For the maximum entropy scheme, there are two types of boundary points as illustrated in Fig. 3. The probability of the two-dimensional random walk being absorbed at point A is $\binom{n-1}{w^*-1} p^{w^*}(1-p)^{n-w^*}$, since the walk must pass through point B. The probability of the two-dimensional random walk being absorbed at point C is $\binom{n}{w^*} p^{w^*}(1-p)^{n-w^*}$, since point C can be approached either through point D or point E. Note that point C represents the initial boundary point in some boundary point set. Let $\{w_i^*\}$ be the set of initial points. Therefore,

$$\left(\frac{\bar{N}}{n}\right) L = N\sum_{n=n_1}^{2^S}\left(\frac{1}{n}\right)\binom{n-1}{w^*-1} p^{w^*}(1-p)^{n-w^*} + N\sum_{w_i^*}\left(\frac{I}{n}\right)\binom{n-1}{w^*} p^{w^*}(1-p)^{n-w^*}, \quad (8)$$

where n_1 is the first row containing a boundary element.

The maximum and minimum values of average per letter coding rate for the three schemes can also be found.

For run length coding

$$\left(\frac{\bar{N}}{n}\right)^R_{\max} = \lim_{p\to 1/2}\left(\frac{\bar{N}}{n}\right)^R \leqq \left(\frac{N}{n}\right)^R_{\max} = \frac{N}{2} \quad (9)$$

$$\left(\frac{\bar{N}}{n}\right)^R_{\min} = \lim_{p\to 0}\left(\frac{\bar{N}}{n}\right) = \left(\frac{N}{n}\right)_{\min} = \frac{N}{2^{N-1}}, \quad (10)$$

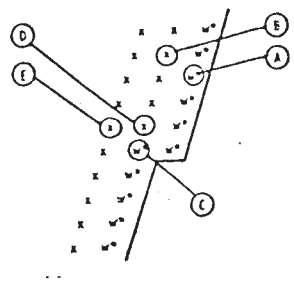

Fig. 3. Illustrating two types of boundary elements.

since, as p approaches zero, all the probability is concentrated in the all zero run of length 2^{N-1}.

For LD coding

$$\left(\frac{\bar{n}}{N}\right)_{\max}^{\mathrm{LD}} = \lim_{p\to 1/2}\left(\frac{\bar{n}}{N}\right)_{\max}^{\mathrm{LD}} = \frac{N+\log_2 N+2}{N} \simeq 1+\frac{\log_2 N}{N} \quad (11)$$

$$\left(\frac{\bar{n}}{N}\right)_{\min}^{\mathrm{LD}} = \lim_{p\to 0}\left(\frac{\bar{n}}{N}\right)^{\mathrm{LD}} = \left(\frac{n}{N}\right)_{\min}^{\mathrm{LD}} = \frac{\log_2 N+2}{N} \simeq \frac{\log_2 N}{N}. \quad (12)$$

For our scheme

$$\left(\frac{\bar{N}}{n}\right)_{\max}^{L} = \lim_{p\to 1/2}\left(\frac{\bar{N}}{n}\right)^{L} \le \left(\frac{N}{n}\right)_{\max}^{L} = \frac{S+\log_2 S}{S-1} \simeq 1+\frac{\log_2 S}{S} \quad (13)$$

$$\left(\frac{\bar{N}}{n}\right)_{\min}^{L} = \lim_{p\to 0}\left(\frac{\bar{N}}{n}\right)^{L} = \left(\frac{N}{n}\right)_{\min}^{L} = \frac{S+\log_2 S}{2^S} = \frac{1}{2^{S-1}}+\frac{\log_2 S}{2^S}. \quad (14)$$

It can be seen that the performance of run length coding is superior to the other two schemes as p approaches zero, although our scheme is practically as good. For $p \gtrsim 0.08$, the LD scheme is the best, although it outperforms our scheme only slightly. Between these two extremes, the maximum entropy scheme is superior to the others.

VII. Remarks

There are two simple ways to implement the maximum entropy coding scheme. One is to store the necessary elements of Pascal's triangle in a read-only memory. Another is to compute the elements a row at a time. This computation requires only one addition per element.

The algorithm can be extended to the M-ary case as Schalkwijk's results indicate. A version of the maximum entropy source coding scheme has been developed and implemented in hardware at the Naval Electronics Laboratory Center at San Diego, CA. It has been applied to the processing of pictorial data [7]. Data transformation techniques were necessary in order to convert the actual source, which is nonbinary and not memoryless, into an approximation of a binary memoryless source. The scheme also seems promising for speech compression.

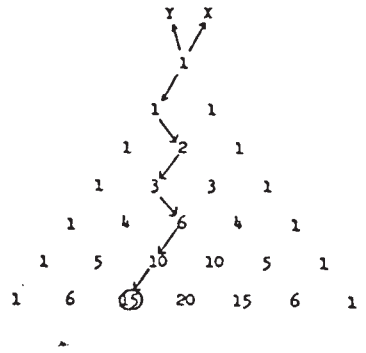

Fig. 4. Schalkwijk block-to-variable coding using Pascal's triangle.

Appendix A

Consider the example of Fig. 4. Taking the X and Y directions as shown, the coding process proceeds as follows. If the incoming source bit is a zero, take one step in the $-X$ direction. If the incoming source bit is a one, take one step in the $-Y$ direction and add to the running sum the number one step in the direction from this point. For example, with reference to Fig. 4, if the block length is $N = 6$ and the incoming source sequence is $010100\cdots$, the running sum would be generated as follows. Since the first bit is a 0, take one step in the $-X$ direction to 1. The second bit is a 1; take one step in the $-Y$ direction to 2 and add to the running sum the number one step in the X direction from this point, which is 1. The third bit is a 0; take one step in the $-X$ direction to 3. The fourth bit is a 1; take one step in the $-Y$ direction to 6 and add the number one step in the X direction from this point, which is 3, to the running sum giving $3 + 1 = 4$. The last two bits are zeroes. Take two steps in the $-X$ direction first to 10 and then to 15, which is the termination point for the encoding process. This point is also the starting point for the decoding process. Therefore, it will be referred to as the "starting point."

The decoding procedure is the inverse of the encoding procedure. Beginning at the starting point, if the running sum is less than the number one step in the X direction, take one step in the X direction and record a 0. If the running sum is greater than or equal to the number one step in the X direction, substract that number from the running sum, record a 1, and move one step in the Y direction. The process will terminate at the apex. Since the last bit encoded was the first bit decoded, the decoded sequence will have to be reversed to obtain the original sequence.

For example, with reference to Fig. 4 again, starting at 15, the running sum 4 is less than the number one step in the X direction which is 10. Take one step in the X direction to 10 and record a 0. The running sum, 4, is less than the number 6, which is one step in the X direction from 10. Therefore, move one step in the X direction to 6 and record a 0. The running sum, 4, is greater than the number one step in the X direction from 6 which is 3. Therefore, subtract 3 from 4 leaving 1, move one step in the Y direction to 3 and record a 1. The running sum, 1, is less than the number one step in the X direction from 3. Therefore, move one step in the X direction to 2 and record a 0. The running sum, 1, is not less than the number one step in the X direction from 2, which is 1. Therefore, subtract 1 from the running sum leaving 0, move one step in the Y direction to 1 and record a 1. The run-

ning sum, 0, is less than the number one step in the X direction from the current starting point, 1. Therefore, move one step in the X direction reaching the apex and record a 0. This completes the process. The decoded bits are 001010 which must be reversed to obtain the original bit sequence.

Appendix B

Proof of Theorem 1: We now proceed to show that as $N \to \infty$, the maximum entropy scheme achieves a coding rate approaching $H(p)$. We consider just the left half of Pascal's triangle. Let (n,ω^*) be an element on the boundary. Then, we know from the definition that $\log_2 \binom{n}{\omega^*} \le S < \log_2 \binom{n+1}{\omega^*+1}$, where n is the length of the source run and the block length $N = S + \log_2 S + 2$. Adding $\log_2 S + 2$ to the above inequality, we have

$$\log_2 \binom{n}{\omega^*} + \log_2 S + 2 \le S + \log_2 S + 2 < \log_2 \binom{n+1}{\omega^*+1} + \log_2 S + 2. \quad (15)$$

We note that $\log_2 \binom{n+1}{\omega^*+1} = \log_2 (n+1) - \log_2 (\omega^*+1) + \log_2 \binom{n}{\omega^*}$. Substituting this in (15) and dividing by n, we have $n^{-1} \log_2 \binom{n}{\omega^*} + n^{-1} \log_2 S + 2 \le N/n < n^{-1} \log_2 (n+1) - n^{-1} \log_2(\omega^*+1) + n^{-1} \log_2 \binom{n}{\omega^*} + n^{-1}(\log_2 S + 2)$. Using Stirling's approximation, it can be shown that $nH(\omega^*/n) - \log_2 (\omega^*+1) - \log (n-\omega^*+1) - \log \sqrt{2\pi} < \log_2 \binom{n}{\omega^*} < nH(\omega^*/n) + \log_2 (n+1)$. Substituting this in (15), we have $H(\omega^*/n) - n^{-1} \log_2 (n-\omega^*+1) - n^{-1} \log_2 \sqrt{2\pi} + n^{-1} \log_2 S - n^{-1} \log_2 (\omega^*+1) + 2/n < N/n < H(\omega^*/n) + 2n^{-1} \log_2 (n+1) + n^{-1} \log_2 S - n^{-1} \log_2 (\omega^*+1) + 2/n$. We make use of the following inequalities

$$n^{-1} \log_2 (n-\omega^*+1) < n^{-1} \log (n-\omega^*+2) < n^{-1} \log_2 (n-\omega^*) + n^{-1} \log_2 2,$$

$$n^{-1} \log_2 (\omega^*+1) < n^{-1} \log_2 (\omega^*+2) < n^{-1} \log_2 \omega^* + n^{-1} \log_2 2,$$

$$n^{-1} \log_2 (n-\omega^*) + n^{-1} \log_2 (\omega^*) < 2n^{-1} \log_2 (n/2) = 2n^{-1} \log_2 n - 2/n,$$

$$\log_2 \sqrt{2\pi} < \log_2 4 = 2.$$

Using the above inequalities, we arrive at the expression

$$H\left(\frac{\omega^*}{n}\right) - n^{-1} \log_2 n^2 + n^{-1} \log_2 S < N/n < H\left(\frac{\omega^*}{n}\right) + 2n^{-1} \log_2 (n+1) + 2n^{-1} \log_2 (n+1) + n^{-1} \log_2 S + n^{-1} \log_2 2.$$

Simplifying further, we get

$$H\left(\frac{\omega^*}{n}\right) - \frac{\log_2 (n^2/S)}{n} < \frac{N}{n} < H\left(\frac{\omega^*}{n}\right) + \frac{\log_2 [2(n+1)^2 S]}{n}.$$

Since $S - 1$ is a lower bound for n and

$$\lim_{N\to\infty} S = \infty,$$

we have

$$\lim_{N\to\infty} n = \infty$$

and

$$\lim_{N\to\infty} \frac{\log n}{n} = 0.$$

Therefore

$$\lim_{N\to\infty} \frac{N}{n} = \lim_{N\to\infty} H\left(\frac{\omega^*}{N}\right).$$

However

$$\lim_{N\to\infty} H\left(\frac{\omega^*}{N}\right) = H(p)$$

since

$$\lim_{N\to\infty} \frac{\omega^*}{n} = p.$$

Therefore,

$$\lim_{N\to\infty} \frac{N}{n} = H(p) \quad \text{wp1}.$$

We have shown that the per letter coding rate approaches the optimal value as the block length approaches infinity, regardless of the source statistics.

References

[1] T. J. Lynch, "Sequence time coding for data compression," *Proc. IEEE*, vol. 54, pp. 1490–1491, Oct. 1966.

[2] L. D. Davisson, "Comments on 'Sequence time coding for data compression,'" *Proc. IEEE*, vol. 54, p. 2010, Dec. 1966.

[3] J. P. M. Schalkwijk, "An algorithm for source coding," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 395–399, May 1972.

[4] L. D. Davisson, "Comments on 'An algorithm for source coding,'" *IEEE Trans. Inform. Theory*, vol. IT-18, p. 827, Nov. 1972.

[5] ——, "Universal noiseless coding," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 783–795, Nov. 1973.

[6] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp 379–423, 623–656, 1948.

[7] J. C. Lawrence, "Application of Schalkwijk source coding techniques to pictorial sources," Naval Electronics Laboratory Center Technical Report 1867, San Diego, CA, 12 March 1973.