LABORATORY FOR
COMPUTER SCIENCE

MASSACHUSETTS
INSTITUTE OF
TECHNOLOGY

MIT/LCS/TM-301

INTERVAL AND RECENCY-RANK SOURCE CODING: TWO ON-LINE ADAPTIVE VARIABLE-LENGTH SCHEMES

PETER ELIAS

APRIL 1986

545 TECHNOLOGY SQUARE, CAMBRIDGE, MASSACHUSETTS 02139

LABORATORY FOR
COMPUTER SCIENCE

MASSACHUSETTS
INSTITUTE OF
TECHNOLOGY

545 TECHNOLOGY SQUARE, CAMBRIDGE, MASSACHUSETTS 02139

INTERVAL AND RECENCY-RANK SOURCE CODING: TWO ON-LINE ADAPTIVE VARIABLE-LENGTH SCHEMES

Peter Elias[1]

ABSTRACT

In these schemes the encoder maps each message into a codeword in a prefix-free codeword set. In interval encoding the codeword is indexed by the interval since the last previous occurrence of that message, and the codeword set must be countably infinite. In recency rank encoding the codeword is indexed by the number of distinct messages in that interval, and there must be no fewer codewords than messages. The decoder decodes each codeword on receipt. Users need not know message probabilities but must agree on indexings, of the codeword set in an order of increasing length and of the message set in some arbitrary order. The average codeword length over a communications bout is never much larger than the value for an off-line scheme which maps the jth most frequent message in the bout into the jth shortest codeword in the given set, and is never too much larger than the value for off-line Huffman encoding of messages into the codeword set best for the bout message frequencies. Both schemes can do much better than Huffman coding when successive selections of each message type cluster much more than in the independent case.

---

1. Department of Electrical Engineering and Computer Science and Laboratory for Computer Science, MIT.

Mail address: Peter Elias, Room 317,
545 Technology Square,
Cambridge, Ma. 02139.

I. INTRODUCTION AND HISTORY

I analyse two on-line adaptive variable-length source coding schemes, interval and recency rank encoding. In each scheme the encoder encodes the current message selected from a set A of m messages into a codeword selected from a prefix-free set C of sequences in an alphabet B of b letters, say B = {0,1,...,b-1}, concatenating the result onto previous output. In interval encoding the codeword represents the interval since the last previous occurrence of the current message, and C must be countably infinite. In recency rank encoding the codeword represents the number of distinct messages present in that interval, and C must be the size of A. The decoder receives the concatenation and decodes each message as soon as the last symbol of its codeword is received. Either scheme can be used by encoders and decoders who know the sets C and A and have agreed to indexings of $C = \{c_1,\ldots,c_n\}$ in an order of increasing length and of $A = \{a_1,\ldots,a_m\}$ in some standard order, but know nothing about probabilities of messages. When messages are selected from a probability distribution on A, the average codeword length for either scheme is never much larger than for probability rank encoding, which maps the jth most probable message in A into the jth shortest codeword in C, and never too much larger than for Huffman encoding, which is probability rank encoding into the codeword set optimal for the given distribution. Both adaptive schemes can do much better than Huffman coding when successive selections of each message type cluster much more than in the independent case.

If users know nothing about message probabilities, the nonadaptive on-line code which is best in a minimax sense has m codewords of length ⌊log(m)⌋, and does relatively poorly in encoding any source whose distribution has entropy much less than log(m). Therefore an on-line scheme which does reasonably well for such sources must be adaptive. Interval and recency rank encoding seem to be the simplest on-line adaptive schemes: simple to describe and to implement, and simple enough to analyze to allow quantitative bounds on their performance in encoding an arbitrary finite sequence. They are therefore pedagogically attractive: I gave homework problems on interval encoding at MIT in 1979, and on both schemes at Harvard in 1984. When this paper was ready for submission I learned that Bentley, Sleator, Tarjan and Wei in [1] had independently introduced recency rank encoding, which they call the move-to-front scheme, at a meeting whose transactions had just appeared. They prove essentially that part of Theorem 1 below which deals with recency rank encoding, and that part of Theorem 2 which shows that the same recency rank result holds for sources which do not have defined probability distributions, when probabilities are replaced by frequencies. The part of Theorem 1 dealing with interval encoding, and the tighter bounds on recency rank encoding and frequency rank encoding in Theorem 2, seem to be new. Both schemes are of interest: recency rank encoding takes fewer bits (in the worst case, for most distributions), but interval encoding has a very simple implementation: it could encode and decode selections from a million-letter alphabet at a microsecond rate.

Faller [8] and Gallager [9] give an elegant on-line adaptive version of Huffman coding, developed further by Knuth [12]. These schemes deal with redundancy due to a nonuniform first order source distribution when the users can generate their own codeword set. They can integrate over an arbitrarily long message sequence and, when message probabilities are well defined, converge to the performance of a Huffman code designed knowing those probabilities. They will then do better than interval or recency rank encoding for messages selected independently from A with those probabilities, but will do worse when selections of each message type cluster in time. Vitter [20] gives a modification with better worst-case results on arbitrary sequences of input symbols.

The Ziv and Lempel schemes [13], [23], [24], [25] are much more sophisticated and powerful, encoding message sequences of increasing length and coping with redundancies of higher order. Modified and augmented versions of the Ziv and Lempel schemes have been implemented by Miller and Wegman [15] and Welch [21], and seem to work very well for some sources, but analytic bounds are less simple. A variant by Rodeh, Pratt and Even [18] which uses universal codewords as variable-length backpointers is close in spirit to interval encoding but preserves the power (and complexity) of dealing with higher order source properties. Other adaptive back-pointer schemes are referenced and analyzed in Gonzalez-Smith and Storer [10].

The work [1] by Bentley et al derives from a scheme for managing a self-organizing file system described and analyzed by Knuth [11]

and (with generalizations) by Rivest [17] in the independent case. Sleator and Tarjan [19] consider the generalized schemes without the independence assumption. If the most recently used item is put at the front of the list, the position of each item is its recency rank. Sending the codeword whose rank in order of increasing length is the message rank can also be done using Rivest's interchange ranking scheme, in which after each access the accessed item and the one which precedes it by k in rank order interchange their ranks. The corresponding coding scheme does better for independent messages but worse in the worst case than recency rank encoding. [17] and [19] give further bibliography of the self-organizing file problem.

II. ON-LINE CODING OF WEAK-LAW SOURCES

Huffman and probability rank encodings can be used on-line only to encode the output of a source which selects messages from A with probabilities that are well-defined, at least in the sense of the weak law of large numbers, and known to the users. Given such a source whose jth most probable message has probability Q(j), users who know message probabilities and are free to choose their own codeword set can use the Huffman algorithm to find an optimal set C_Q for Q, whose jth shortest codeword is $L_Q(j)$ letters long, and do probability rank encoding by assigning that codeword to the jth most probable message in A. The expected codeword length of the resulting Huffman encoding is known to be bounded above and below in terms of the entropy H(Q):

$$(2.1) \qquad H(Q) = \Sigma\, Q(j)\log(1/Q(j)),$$

$$\max\{1,H(Q)\} \leq E_Q(L_Q) = \Sigma\, Q(j)L_Q(j) < H(Q) + 1.$$

(The logarithmic base is the alphabet size b. Expectations are summed over all and only terms with positive probabilities.)

On-line Huffman coding is not available to users who do not know their message probabilities exactly, or cannot choose their own codeword sets. If users must map their message set 1-1 into a fixed prefix-free codeword set C given in advance, then probability rank encoding is optimal for users who know message probabilities, or atleast know a ranking of their messages in an order of decreasing probability. The given set C may be chosen to do reasonably well for all distributions in some class. If there

is no upper bound to the number of positive probabilities in the distribution, an infinite set C is required. One such set, C_1, has as its jth codeword $c_1(j)$ the usual $(1 + \lfloor \log(j) \rfloor)$-symbol b-ary representation of the integer j, prefixed by a sequence of $\lfloor \log(j) \rfloor$ 0's. Another such set, C_2, has as its jth codeword the b-ary representation of j preceded by $c_1(1+\lfloor \log(j) \rfloor)$. The lengths $L_1(j)$, $L_2(j)$ of these codewords are then

$$(2.2)\quad \begin{aligned} L_1(j) &= 1 + 2\lfloor \log(j) \rfloor \\ &\le 1 + 2\log(j) = G_1(\log(j)), \\ L_2(j) &= 2 + \lfloor \log(j) \rfloor + 2\lfloor \log(1+\lfloor \log(j) \rfloor) \rfloor \\ &\le 2 + \log(j) + 2\log(1+\log(j)) = G_2(\log(j)), \end{aligned}$$

for $j \ge 1$, and the expected codeword lengths $E_Q(L_1)$, $E_Q(L_2)$ are bounded above respectively by

$$(2.3)\quad \begin{aligned} \Sigma\, Q(j)(1+2\log(j)) &\le 1+2\Sigma\, Q(j)\log(1/Q(j)) = 1+2H(Q) \\ &= G_1(H(Q)), \\ \Sigma\, Q(j)(2+\log(j)+2\log(1+\log(j))) &\le 2+H(Q)+2\log(1+H(Q)) \\ &= G_2(H(Q)), \end{aligned}$$

using the fact that $G_1(v)$ and $G_2(v)$ in (2.2) are convex ∩ and increasing in v, and the Wyner inequality $E_Q(\log) \le H(Q)$ [22], which holds since Q is nonincreasing so $1/Q(j) \ge j$. Thus knowing only ranking, not Q, and doing probability rank encoding into C_1 or C_2 rather than into the optimal C_Q, costs a factor in average codeword length which is at most 3 for C_1, and for C_2 approaches 1 for large H(Q). Infinite prefix-free codeword sets were discussed by Levenstein in [14]. Results like (2.3) for a set like C_1 were given in [4], and in

[5] for C_1 and a binary version of C_2. (For b=2 the function L_2 can be improved to $L_2'(j) = 1+\lfloor\log(j)\rfloor+2\lfloor\log(1+\log(j))\rfloor$, which satisfies the Kraft inequality with equality. Other sets better for very large values of H are described in [5], [2] and [7].) Algorithms for constructing a C which is minimax optimal for probability rank encoding -- i.e. which minimizes over some class of distributions the maximum of some measure comparing expected codeword length to expected Huffman codeword length -- are given in [3], [16] and [6].

Probability rank encoding into a minimax-optimal codeword set is sensible for clients of a multiuser system who each know the probabilities of their messages but must share a single system-wide codeword set. Users who do not know probabilities, however, are common in data processing. Such users are not likely to know an accurate probability ranking of their message set, so they cannot do accurate probability rank encoding on line. The work reported here started from the observation that interval coding into C_1 or C_2, which is suitable for such users because it requires no knowledge of either Q or ranking, has an average codeword length which also satisfies the upper bounds in (2.3).

In interval coding into C_1 or C_2, assume that all messages have been encoded and decoded correctly at least once by time t. (Initialization is dealt with in Section 3.) The coder sends the kth codeword c_k at time t if the message a_i just selected was most recently selected at time t-k. By the weak law of large numbers the relative frequency of the jth most probable message among the first N messages approaches Q(j) in probability as N

increases, so the mean interval between occurrences of that message approaches 1/Q(j). The upper bounding functions $G_1(\log(u))$, $G_2(\log(u))$ in (2.2) are convex $\cap$ in u, so their values at the mean argument $u = 1/Q(j)$ are greater than the limits of the means of L_1 and L_2 over the different intervals between successive occurrences of the jth most probable message. Thus the mean lengths of the codewords which represent the jth most probable message are bounded above in probability by $G_1(\log(1/Q(j)))$ and $G_2(\log(1/Q(j)))$ respectively. The mean lengths of all codewords are therefore bounded above in probability by the expectations of those quantities with respect to Q(j). Since G_1 and G_2 are convex $\cap$ in $v = \log(u)$, that expectation is also bounded by the right sides of (2.3).

In recency rank encoding the coder sends c_k at time t to represent the present message a_i if k-1 other message types have occurred since the most recent previous occurrence of a_i. The interval since the last occurrence of a message will be larger than its recency rank if some other message occurs more than once in that interval. Since larger integers are mapped into longer codewords, recency rank encoding never takes more and sometimes takes fewer symbols per message than interval encoding into the same C, so its average codeword length over a long sequence is bounded in probability by (2.3) again. Its possibly greater economy in channel use is balanced by its need for more memory accesses to encode a message: see the algorithms in Section 3 below.

Theorem 1 summarizes the obvious generalization of the above

results for on-line encoding of sources with defined probabilities. Theorem 2 gives more precise and general bounds on average codeword lengths for on-line interval and recency rank encodings of an arbitrary finite sequence of messages, with no assumptions about the character of its source, and compares their performance to that of off-line adaptive probability rank and Huffman encodings. The tighter bounds in Theorem 2 show differences in performance between interval, recency rank and probability rank encodings which persist in the limit when messages have defined probabilities, while Theorem 1 gives the same upper bound for all three. Bentley et al [1] give upper bounds for recency rank encoding similar to those in Theorem 1 for both independent letter sources (their Theorem 2) and for arbitrary sequences (their Theorem 1).

III. BOUNDS FOR WEAK-LAW SOURCES.

Some definitions are needed for Theorem 1.

A function $f:A^*\to B^*$ mapping message sequences into sequences of channel symbols is an on-line scheme if it is 1-1 and

(i) $f(\emptyset)=\emptyset$,

(ii) for each $t > 0$ and $\alpha(1),\ldots,\alpha(t)$ in A^t the set $\{f(\alpha(1),\ldots,\alpha(t),a_i)\,|\,1\le i\le m\}$ of m sequences in B^* have the common prefix $f(\alpha(1),\ldots,\alpha(t))$, and

(iii) deleting that common prefix leaves a prefix-free set of m codewords in B^*.

An on-line scheme is nonadaptive if $f(\alpha(1),\ldots,\alpha(t))$ is the concatenation of $f(\alpha(1)),\ldots,f(\alpha(t))$, and otherwise is adaptive.

Let $F(A,B)$ denote the class of all on-line schemes from A^* to B^*.

A source s is a weak law source with probability rank distribution Q if in its first N selections $\alpha(1),\ldots,\alpha(N)$ it chooses the jth most probable message with a relative frequency which converges in probability to $Q(j)$ as N grows. Let $S(Q)$ denote the class of all such sources.

The mean cost in b-ary symbols per message of encoding the first N

messages from a source $s \in S(Q)$ using the coding scheme $f \in F(A,B)$ is the quotient $|f(\alpha(1),\ldots,\alpha(N))|/N$, where $|X|$ denotes the number of symbols in the string X. That quotient need not approach a limit in probability as N increases, but will have limits superior and inferior in probability. Let

$$M_{sup}(f,s) = \operatorname*{plimsup}_{N\to\infty} |f(\alpha(1),\ldots,\alpha(N))|/N,$$

$$M_{inf}(f,s) = \operatorname*{pliminf}_{N\to\infty} |f(\alpha(1),\ldots,\alpha(N))|/N.$$

If a limit in probability in fact exists, denote it by

$$M(f,s) = M_{inf}(f,s) = M_{sup}(f,s).$$

THEOREM 1

Let Q be a nonincreasing distribution on the first m positive integers. Let L_Q be the length function of a Huffman codeword set for Q as in (2.1). Let L be the length function of a countably infinite prefix-free codeword set which, like L_1 and L_2 in (2.2), has an upper bound

$$L(j) \leq G(\log(j)),$$

where G(v) is convex $\cap$ and nondecreasing in v, so G(log(u)) is convex $\cap$ and nondecreasing in u. Let Huffman coding into C_Q and probability rank, interval and recency rank encodings into C be represented respectively by the online coding functions f_H,

f_{PR}, f_{IN} and f_{RR} in F(A,B). Then for all s∈S(Q),

(i)

$$\sup_{s'\in S(Q)} \inf_{f\in F(A,B)} M_{inf}(f,s') = E_Q(L_Q) = M(f_H,s) \geq \max\{1,H(Q)\},$$

$$< 1 + H(Q).$$

(ii)

$$M(f_{PR},s) = E_Q(L) \geq \max\{L(1),H(Q)\},$$

$$\leq G(H(Q)).$$

(iii)

$$\lfloor f_{IN}(\alpha(1),\ldots,\alpha(N))\rfloor/N \geq \lfloor f_{RR}(\alpha(1),\ldots,\alpha(N))\rfloor/N \geq L(1),$$

$$M_{sup}(f_{RR},s) \leq M_{sup}(f_{IN},s) \leq G(H(Q)).$$

Proof

The fact that $M(f_H,s)$ and $M(f_{PR},s)$ are constant on S(Q) and equal to the expectations of L_Q and L is immediate from the definition of a weak law source and the weak law of large numbers itself. The rest of (i) is standard. The upper bound in (ii) was proved above in (2.3). The lower bound L(1) in (ii) and (iii) holds since an on-line encoder is 1-1, so every message is mapped into some codeword, which is no shorter than the shortest. The proof of the upper bound in (iii) is the immediate generalization of the proof given above that the bounds in (2.3) hold for interval, and thus *a fortiori* for recency rank, encodings. A more formal proof of Theorem 1 results from taking appropriate limits in probability in the more detailed Theorem 2 below. ∎

Comments on Theorem 1

The mean codeword lengths of Huffman and probability rank encoding

have limits in probability which are constant on S(Q). The mean codeword length of an adaptive encoding of the output of an s in S(Q) need not approach a limit in probability, and if it does the limit need not be constant on S(Q). The upper and lower bounds in (iii) may therefore differ widely, unlike those in (i) and (ii). Both bounds are approached or attained as limits in probability by some sources in S(Q) for some Q and some codeword sets C.

For example let Q_m be uniform, $Q_m(i) = 1/m$ on A, with $m=b^k$ for integer k. The cyclic source $s_n \in S(Q_m)$ which runs through messages in standard order generating n successive copies of each has period nm. After the first input cycle, for n = 1 the output of s_1 is mapped by both interval and recency rank encoders into a sequence of copies of the mth shortest codeword. That gives mean codeword length $L_1(m) = 1 + 2k$ for C_1, which attains the upper bound $1 + 2H(Q_m)$ for any k, and $L_2(m) = 2+k+2\lfloor \log(1+k) \rfloor$ for C_2, which attains the upper bound $2+H(Q_m)+2\log(1+H(Q_m))$ when $k = b^{j-1}$ for some integer j. For $n > 1$ the outputs of both interval and recency rank encodings of s_n are cycles of n codewords starting with the nmth or mth, followed by n-1 copies of the shortest codeword. Since $L(j) < k\log(j)$ for some k, mean codeword length approaches L(1) for both schemes as n grows.

The coding scheme corresponding to Rivest's interchange rule for files [17], in which each message has an associated rank at time t and when selected interchanges ranks with the message whose rank is k smaller, does not satisfy the upper bound in Theorem 1 on all of $S(Q_m)$ for large m. It always sends the mth longest codeword if the message of rank m always occurs next. That situation

occurs if the only two of the m messages which actually appear are the two which have been assigned initial ranks m and m-k, and those two alternate forever. In that case H(Q) is log(2) b-ary symbols, the upper bound for C_1 from Theorem 1 is $1+2\lfloor \log_b(2) \rfloor$ (3 if b = 2 and 1 if b > 2) and is attained by both interval and recency-rank encoding into C_1, while the mean codeword length using Rivest's algorithm is $\geq \log(m)$ for any prefix-free C with m members, and is $1+2\log(m)$ for C_1.

The fact that a source is in S(Q) does not guarantee the existence or determine the value of its limiting mean codeword length under interval or rank encoding into some given C. However the existence of a stationary conditional distribution of the intervals or recency ranks of the jth most probable message for each j does. As an important example let s(Q) denote the source in S(Q) which picks messages independently from Q. If the jth most probable message occurs at time t, the conditional probability that it last occurred at time t-k is just $Q(j)(1-Q(j))^{k-1}$. Averaging L(k) over positive k gives a mean codeword length for the jth most probable message. Averaging over j then gives

$$(3.1) \qquad M(f_{IN},s(Q)) = \sum_j Q^2(j) \sum_k (1-Q(j))^{k-1} L(k).$$

Formulas for computing the conditional distribution of the recency rank of message j when it is selected by s(Q) are available but complex [11]. However there are simple closed forms for the mean recency rank of the jth most probable message and of all messages from s(Q) (e.g. in [17]), which give not an exact value like

(3.1) but two additional convexity bounds,

$$(3.2)\quad M(f_{RR},s(Q_m)) \le \sum_i Q(i)G\Big(1/2 + \sum_j Q(j)/(Q(i)+Q(j))\Big),$$
$$\le G\Big(1/2 + \sum_{i,j} Q(i)Q(j)/(Q(i)+Q(j))\Big),$$

which may or may not be tighter than the bound G(H(Q)) in the Theorem. When $Q = Q_m$, the uniform distribution, the recency rank of each message also has a uniform distribution. Then an exact expression is immediate:

$$(3.3)\qquad M(f_{RR},s(Q_m)) = (1/m)\sum_{k=1}^{m} L(k).$$

On the whole the independent letter source in S(Q) has a limited role here, since it is not extremal in its performance as it is in so many information theory problems: in general it attains neither bound in (iii) for interesting L.

IV. ENCODING ARBITRARY MESSAGE SEQUENCES.

Next, consider encoding an arbitrary sequence $\alpha(1),\ldots \alpha(N)$ of N selections from $A = \{a_1,\ldots,a_m\}$. To intitialize the adaptive schemes extend the sequence on the left by setting

(4.1) $\quad \alpha(t) = a_i$ at $t=1-i$, $1 \leq t \leq m$,

as though all m messages had been sent just before the start of actual transmission, in the inverse of the users' standard order. Let T denote the set of N transmission times, T(i) the set of times at which message a_i is selected, n(i) the number of such selections, $t_k(i)$ the time of the kth such selection, R(i) the relative frequency and r(i) the rank in an ordering by decreasing frequency of message a_i, and Q(j) the relative frequency of the message of rank j = r(i). The permutation r is not unique if there are equiprobable messages, but the value of Q(j) is. (4.2) defines these terms and one such r for reference.

(4.2)
$$T = \{1,\ldots,N\}, \quad T(i) = \{t \text{ in } T \mid \alpha(t) = a_i\},$$
$$n(i) = |T(i)|, \quad R(i) = n(i)/N,$$
$$t_0(i) = 1-i,$$
$$t_k(i) = \min \{t \in T(i) \mid t > t_{k-1}(i)\}, \ 1 \leq k \leq n(i),$$
$$r(i) = |\{i' \mid R(i') > R(i), \text{ or } R(i') = R(i) \text{ and } i' \leq i\}|,$$
$$Q(r(i)) = R(i),$$

where $t_0(i)$ comes from (4.1) (and is <u>not</u> a member of T(i)).

Interval and recency rank encodings can be defined as on-line schemes on an arbitrary sequence in A^N once the indexed set C

and the standard order on A needed in (4.1) are given. Huffman and other probability rank encodings are not defined as on-line schemes when no probabilities are given in advance. The appropriate schemes for comparison are off-line adaptive or two-pass schemes, which wait until all N messages are available, compute the relative frequencies and treat them as probabilities. The frequency rank function r in (4.2) defines an off-line frequency rank encoding into any C, and an off-line adaptive Huffman encoding is just frequency rank encoding into the set C_α optimal for the Q in (4.2). Such off-line adaptive Huffman encoding is used for the compaction of large files.

All these encoders, on-line and off-line, can be decomposed into two stages. The first stage maps the message sequence $\alpha(1),\ldots,\alpha(N)$ into a sequence of integers. Let $x(t)$, $y(t)$ and $z(t)$ denote respectively the outputs of the first stages of interval, recency rank and frequency rank encoders at time t. Then for $t\in T$,

$$x(t) = \min\{k \mid \alpha(t-k) = \alpha(t)\},$$
$$y(t) = |\{\alpha(t') \mid t-x(t) < t' \leq t\}|, \tag{4.3}$$
$$z(t) = r(\alpha(t)).$$

Because of (4.1) the sets on the right in the first two lines of (4.3) are never empty for $t\in T$.

If the message set is the first m positive integers and the standard order is the usual order on integers then message i is its own index, $A = \{1,\ldots,m\}$, and these first stages of coders map integers to integers. Then simple algorithms compute outputs $x(t)$ and $y(t)$ at time t given input $\alpha(t)$. A sample implementation

of interval encoding distributes the initialization effort, which is useful if m is very large. It uses an m-element array LAST of integers $\leq$ N initialized to 0.

INTERVAL ENCODING

```
for t=1 to N do
   begin
        if LAST(α(t))=0 then x(t)=t+α(t)-1
                        else x(t)=t-LAST(α(t));
        LAST(α(t)) = t
   end
```

This algorithm takes O(1) operations per input integer. Since the m words used to store entries in LAST must be finite an occaisonal reinitialization is required, but a 48 bit wordsize allows a message per microsecond for almost nine years between restarts. The more complex part of an implementation for a large message set whose members are not the integers is computing message indices (addresses) in time O(1), for example by hashing.

A sample implementation of recency rank encoding uses an m-element array RANK of integers $\leq$ m. It initializes first, since its time requirement makes it less appealing for very large m.

RECENCY RANK ENCODING

```
for j=1 to m do RANK(j) = j;
```

```
for t=1 to N do
    begin
        y(t)=RANK(α(t));
        for i=1 to m do if RANK(i)<y(t)
                                then RANK(i)=RANK(i)+1;
        RANK(α(t))=1
    end
```

The recency rank algorithm takes O(m) operations for initialization and for each input. Storing the ranking data as a linked list of messages, linked in order of increasing rank, and using the move-to-front algorithm ([1], [11], [17], [19]), requires instead y(t) accesses to find y(t) and O(1) more to relink it at the head of the list. That is a significant mean improvement if the algorithm is doing significant data compression on average, e.g. if y(t) typically has half as many significant bits as α(t), but access cost still grows like some function of m, depending on distribution assumptions. More imaginative algorithms, using more complex data structures and caching and batching of updates, might do better. An algorithm for Rivest's interchange ranking scheme is much more economical: it maintains two arrays of size m, one giving the rank of message i and the other the index of the message of rank j, and takes O(1) accesses to update both.

The second stage of all three schemes maps the integer j into c_j, the jth codeword in a prefix-free set C of size n indexed in an order of nondecreasing length. The indexing is not unique if there are codewords of equal length, but the length L(j) of the

jth codeword is. C must be countably infinite for interval encoding, and may be no smaller than A for frequency rank or recency rank encoding. The function L must satisfy the Kraft inequality

$$\sum_{i=1}^{n} b^{-L(i)} \leq 1, \tag{4.4}$$

known to be necessary and sufficient for the existence of a prefix-free set C with length function L. The cost of encoding a message $\alpha(t)$ into C is $L(w(t))$, where $w(t)$ is the output of one of the three encoder first stages defined by (4.3). Those costs are bounded next.

V. COST BOUNDS FOR ARBITRARY MESSAGE SEQUENCES

The mean cost per message of encoding N messages into a sequence of codewords in a set C with length function L is the mean over the N messages of the length L(w(t)) of the encoding into C of the output w(t) of the first coding stage just discussed, where w is x or y or z in (4.3). For any function u on T let $\Sigma_T u$ and $\Sigma_{T\langle i\rangle}$ denote sums of u(t) over the t in T and in T(i), respectively, and denote the corresponding means by

(5.1) $M_T(u) = (1/N)\Sigma_T u$, $M_{T\langle i\rangle}(u) = (1/n(i))\Sigma_{T\langle i\rangle} u$.

Then the mean codeword length per message is the expectation of the means over the T(i) with respect to the distribution R(i):

(5.2) $$M_T(L(w)) = (1/N)\sum_i \Sigma_{T\langle i\rangle} L(w) = \sum R(i) M_{T\langle i\rangle}(L(w)).$$

The means in (5.2) can be bounded above by bounding L above by

(5.3) $$L(j) \leq G(\log(j)),$$

where G(v) = G(log(u)) is convex ∩ and nondecreasing in v, and therefore also in u. (The convex hull of $\{(\log(j), L(j)) \mid 1 \leq j \leq n\}$ is the smallest such G, defined for all finite n and for some L with infinite n, like L_1 and L_2. Other G, like G_1 and G_2, are sometimes more convenient.) Bounding L(w) by G(log(w)) term by term on the right in (5.2) and then applying first convexity in w and then convexity in log(w) gives

(5.4) $M_T(L(w)) \leq \Sigma\, R(i)G(\log(M_{T\langle i\rangle}(w))),$

$\leq G(\Sigma\, R(i)\log(M_{T\langle i\rangle}(w)))$

$= G(\Sigma\, R(i)(\log(1/R(i))+\log(R(i)M_{T\langle i\rangle}(w))))$

$= G(H(Q) + \Sigma\, R(i)\log(R(i)M_{T\langle i\rangle}(w))).$

A weaker bound comes from moving the averaging over R(i) to the argument of the logarithm in the second line of (5.4) and using (5.2):

(5.5) $M_T(L(w)) \leq G(\log(M_T(w))).$

Evaluating $M_{T\langle i\rangle}(w)$ and $M_T(w)$ in (5.4) and (5.5) as functions of the rank distribution Q and the parameters m, N when w is x and y and z gives the several upper bounds in Theorem 2.

THEOREM 2

Let $\alpha(1),\ldots,\alpha(N)$ be any sequence in A^N, $A = \{a_1,\ldots a_m\}$. Extend the sequence by (4.1) and define the rank frequencies Q(j) by (4.2), the first stage outputs x(t), y(t) and z(t) of interval, recency rank and frequency rank encoders by (4.3) and the average $M_T(u)$ of u over T by (5.1). Let C_n be a prefix-free set of $n \geq m$ codewords with length function L_n bounded by

(5.6) $L_n(j) \leq G_n(\log(j))$, for $1 \leq j \leq n$,

and, if $n < \infty$, let C be a countably infinite prefix-free codeword set with length function L bounded by

(5.7) $L(j) \leq G(\log(j))$, for integer $j>0$,

where $G_n(.)$ and $G(.)$ are convex $\cap$ and nondecreasing functions of real argument on their respective domains $[0,\log(n)]$ and $[0,\infty)$. Let L_α be the length function of the Huffman codeword set optimal for Q, and let $\delta = (m-1)/N$. Then

(i)
$$M_T(L(x)) \geq L(1),$$
$$\leq G(H(Q) + \log(1+\delta)).$$
$$\leq G(\log(m)+\log(1+\delta/2)),$$

(ii)
$$M_T(L_n(y)) \geq L_n(1),$$
$$\leq G_n(H(Q) + \sum_{k>j} Q(j)\log(jQ(j) + Q(k) + \delta),$$
$$\leq G_n(\log(2E_\alpha(.)-1+\delta m/2)),$$

(iii)
$$M_T(L_n(z)) \geq \max\{L_n(1),H(Q)\},$$
$$\leq G_n(H(Q) + \sum Q(j)\log(jQ(j))),$$
$$\leq G_n(\log(E_\alpha(.))),$$

(iv)
$$M_T(L_\alpha(z)) = E_\alpha(L_\alpha) \geq \max\{1,H(Q)\},$$
$$< H(Q) + 1.$$

Comments.

Some comments and examples show the relation of Theorem 2 to Theorem 1 and illustrate the various upper bounds. The proof of the theorem follows.

For a weak law source, as $N\to\infty$ the upper bounds on sample means become upper bounds on limit superiors in probability, the rank probability distribution is the limit in probability of the rank frequency distribution of the source, and Theorem 2 with $\delta = 0$ can

be read as a more detailed and tighter version of Theorem 1.

The first upper bound in each of (i), (ii) and (iii) comes from (5.4) and is tighter than the second. These bounds get tighter in going from (i) to (iii) because G is an increasing function of its argument and the quantities added to its argument from (i) to (iii) are nonpositive: since Q(j) decreases with j,

$$\log(jQ(j)) \le \log(jQ(j) + \sum_{k>j} Q(k)) \le \log(\sum_{j} Q(j)) = \log(1) = 0.$$

While the second upper bound in each of (i), (ii), (iii) (which comes from (5.5)) is weaker it also has interest since it bounds in terms of expectation rather than entropy. These bounds also get tighter from (i) to (iii): since the mean of a nondecreasing distribution on {1,...,m} is at least 1 and at most (m+1)/2,

$$\log(E_Q(.)) \le \log(2E_Q(.)-1) \le \log(m).$$

For the uniform distribution $Q_m(j) = 1/m$, in the limit of large N the four arguments of G and G_n in the two upper bounds in each of (i) and (ii) all equal log(m). The arguments are a bit smaller in (iii), where the argument is (1/m)log(m!) for the first bound (approaching log(m/e) for large m) and log((m+1)/2) for the second. The examples given in Theorem 1 still approach or attain the upper and lower bounds in (i) and (ii) when $C_n = C = C_1$ or C_2.

For nonuniform distributions, however, the arguments of G in the

six bounds can all be quite different. Let Q be a binary source, m = b = 2, with relative frequencies Q(1) = 0.9, Q(2) = 0.1. In the limit of large N,

$$M_T(L(x)) \leq G(H(Q)) = G(0.4690)$$
$$\leq G(\log(m)) = G(1),$$

$$M_T(L_n(y)) \leq G_n(H(Q)+\Sigma\ Q(j)\log(jQ(j)+\Sigma_{k>j}\ Q(k)) = G(0.2368)$$
$$\leq G_n(\log(2E_\alpha(.)-1)) = G_n(0.2630),$$

$$M_T(L_n(z)) \leq G_n(H(Q) + \Sigma\ Q(j)\log(jQ(j))) = G_n(0.1)$$
$$\leq G_n(\log(E_\alpha(.))) = G_n(0.1375).$$

The entropy bound in (i) can be tighter than the expectation bounds in (ii) and (iii), although that is not illustrated by the last example. E. g. if b = 2, m = 1000, Q(1) = 0.9 and Q(i) = 0.1/999 for all other i, $H(Q)=1.4654 < \log(E_\alpha(.))=5.6710$.

Proof

As in Theorem 1 the Huffman bounds (iv) are standard and all lower bounds are obvious since each coding scheme is 1-1. It remains to prove the upper bounds in (i), (ii) and (iii).

By (4.2) and (4.3) the mean of the frequency rank variable z over T(i) is just the rank r(i) of the constant value which $\alpha(t)$ takes on T(i), if T(i) is not empty. And (5.2) gives the mean over T as the expectation with respect to R(i) of the means over the T(i). With the relation between R and Q from (4.2) this gives

(5.8)

$$\Sigma\, R(i)\log(R(i)M_{T\langle i\rangle}(z)) = \Sigma\, R(i)\log(R(i)r(i)) = \Sigma\, Q(j)\log(jQ(j)),$$

$$M_T(z) = \Sigma\, R(i)r(i) = E_R(r) = \Sigma\, jQ(j) = E_Q(.).$$

And (5.8), with (5.4) and (5.5), proves (iii).

The sum of the interval variable x over T(i), from (4.1), (4.2) and (4.3), is just the interval from $t_o(i)$ to $t_{n\langle i\rangle}(i)$ and is bounded above since $t_{n\langle i\rangle}(i) \leq N$. A bound on the mean over T follows in the third line of (5.9) by (5.2):

$$(5.9)\quad \Sigma_{T\langle i\rangle}\, x(t) = t_{n\langle i\rangle}(i) + i-1 \leq N + i-1,$$

$$R(i)M_{T\langle i\rangle}(x) \leq (1+(i-1)/N),$$

$$M_T(x) = \Sigma\, R(i)\, M_{T\langle i\rangle}(x) \leq m(1 + (m-1)/2N),$$

which with (5.4) and (5.5) proves (i).

To bound the sum over T(i) of the recency rank variable and its average over T, from (4.2), (5.1) and (5.2),

$$(5.10)\quad y(t_j(i)) = |\{\alpha(t)\,|\,0<t\leq t_1(i)\}| + i-1,\ j=1,$$

$$= |\{\alpha(t)\,|\,t_{j-1}(i)<t\leq t_j(i)\}|,\ 1<j\leq n(i),$$

$$\Sigma_{T\langle i\rangle} y(t) \leq \sum_k \min\{n(i),n(k)\} + i-1$$

$$R(i)M_{T\langle i\rangle}(y) \leq \left(\sum_k \min\{R(i),R(k)\} + (i-1)/N\right),$$

$$\leq \left(r(i)R(i) + \sum_{r(k)>r(i)} R(k)\right) + (i-1)/N\ ),$$

$$M_T(y) = \Sigma\, R(i)M_{T\langle i\rangle}(y) \leq \Sigma\, jQ(j) + \sum_j \sum_{k>j} Q(k) + m(m-1)/2N,$$

$$= 2E_Q(.) - 1 + m(m-1)/2N.$$

$$\Sigma R(i)\log(R(i)M_{T\langle i\rangle}(y) < \sum_j Q(j)\log\Big(jQ(j) + \sum_{k>j} Q(k) + (m-1)/N\Big).$$

Summing over $1 \leq j \leq n(i)$ in the first two lines of (5.10) gives the third line, since message k can contribute at most 1 to each of the $n(i)$ values $y(t_j(i))$ in the sum, by appearing as the value of $\alpha(t)$ at one or more times in the jth of the $n(i)$ disjoint intervals on the right in lines 1 or 2, and can contribute to at most $n(k)$ such values in all, and thus can contribute at most $\min\{n(i),n(k)\}$ to the sum in line three. The fifth line follows since the definition of the rank function in (4.2) gives $R(k) \geq R(i)$ when $r(k)>r(i)$. Summing over i, using $R(i) = Q(r(i))$ from (4.2) and changing the summation index to $j = r(i)$ gives the next two lines, and a similar summation and bounding i by m gives the last line. (5.4) and (5.5) complete the proof of (ii). ∎

VI. OPEN PROBLEMS AND COMMENTS

This paper has shown that interval and recency rank encoding into C_1 and C_2 do fairly well relative to Huffman coding for any Q on any M, and that C_2 is asymptotically optimal [5] as H(Q) increases. It has not found the best codeword sets for these schemes. Finding codeword sets which are minimax optimal relative to Huffman encoding, for interval and recency rank encodings, by minimizing the maximum of each of several comparative cost measures over each of various classes of distributions, is a set of open problems similar to but distinct from the several optimization problems solved in [3], [6] and [16].

The analysis has assumed no knowledge of the probability ranking of M by the encoder. Available knowledge can be used to make the standard order in (4.1) as near to probability rank order as possible, which will reduce the starting transient but will not affect long term behaviour.

An interesting question in concrete complexity mentioned earlier is finding an efficient recency rank encoding algorithm for large message sets, and in particular determining whether it is possible to code in a time per letter independent of the sizee m of the message set.

A finite message set has been assumed throughout. The schemes work well with an infinite set in the steady state, giving finite average codeword lengths for C_1 and C_2 if H(Q) is finite, but the steady state is never reached if m is infinite and the

standard order is indeed strictly random with respect to a probability ranking. Under those circumstances for each integer K there is probability 1 that the first message has a rank in standard order which takes more than K b-ary symbols to write, so expected codeword length is infinite even if Q actually assigns positive probabilities only to a finite (but unknown) subset of A. Partial knowledge of ranking is sufficient for convergence, however. For example in encoding the infinite message set of all grammatical English sentences, the initial value for each sentence which is its ASCII character string read as a binary number ensures finite numbers at the output of the first stage of an interval encoder, but reaching steady state takes more than whatever time is available. Finding necessary and sufficient conditions on the relation between an initial ordering of the integers and a probability ranking or a probability distribution on the integers such that expected codeword length is bounded and converges is another interesting open problem.

REFERENCES

[1] J.L. Bentley, D. Sleator, R.E. Tarjan and V.K. Wei, "A locally adaptive data compression scheme", Proceedings of the Twenty-second Allerton Conference on Communication, Control and Computing, pp.233-242, October 1984.

[2] J.L. Bentley and A.C. Yao, "An almost optimal algorithm for unbounded searching", Inf. Processing Letters, vol. 5, no. 3, pp. 82-87.

[3] L.D. Davisson and A. Leon-Garcia, "A source matching approach to finding minimax codes", IEEE Trans. Inform. Theory, vol. IT-26, no. 2, pp. 166-174, March 1980.

[4] P. Elias, "Minimum times and memories needed to compute the values of a function", J. Comput. Syst. Sci., vol. 9, no. 2, pp. 196-212, Oct. 1974

[5] P. Elias, "Universal codeword sets and representations of the integers", IEEE Trans. Inform. Theory, vol. IT-21, no. 2, pp.194-203, March 1975.

[6] P. Elias, "Minimax optimal universal codeword sets", IEEE Trans. Infor. Theory, vol. IT-29, no. 4, pp. 491-502, July 1983.

[7] S. Even and M. Rodeh, "Economical encoding of commas between strings",Comm. Assoc. for Comp. Mach., vol. 21 no. 4, pp. 315-317, April 1978.

[8] N. Faller, "An adaptive system for data compression", Conference Record, Asilomar Conference on Circuits, Systems and Computers, 7th conference, pp. 593-597, 1973.

[9] R.G. Gallager, "Variations on a theme by Huffman", IEEE Trans. Inform. Theory, vol. IT-24, no. 6, pp. 668-674, Nov.

1978.

[10] M.E. Gonzalez Smith and J.A. Storer, "Parallel algorithms for data compression", J. Assoc. Comput. Mach., vol. 32, no. 2, pp. 344-373, April 1985.

[11] D.E. Knuth, The Art of Computer Programming, Vol. 3, Sorting and Searching, pp. 398-399 and excercises 11, 12, p. 403, Addison -Wesley, Reading, Mass. 1973.

[12] D.E. Knuth, "Dynamic Huffman Coding", Journal of Algorithms, vol. 6, no. 2, pp. 163-180, June 1985.

[13] A. Lempel and J. Ziv, "On the complexity of finite sequences", IEEE Trans. Inform. Theory, vol. IT-22, no. 1, pp. 75-81, Jan. 1976.

[14] V.I. Levenstein, "The redundancy and decceleration of a separative encoding of the natural numbers", Probl. Cybern., no. 20, pp.173-179, Moscow, 1968.

[15] V.S. Miller and M. N. Wegman, "Variations on a theme by Lempel and Ziv", to be published in Combinatorial algorithms on words, Apostolico and Galil, editors, Lecture Notes in Computer Science, Springer, Berlin.

[16] J. Rissanen, "Minimax codes for finite alphabets", IEEE Trans. Inform. Theory, vol. IT-24, no. 3, pp.389-392, May 1978.

[17] R. Rivest, "On self-organizing search heuristics", Comm. Assoc. Comp. Mach., vol. 19, no. 2, pp. 63-67, Feb. 1976.

[18] M. Rodeh, V. Pratt and S. Even, "Linear algorithm for data compression via string matching", J. Assoc. for Comp. Mach., vol. 28, no. 1, pp.16-24, Jan. 1981.

[19] D.D. Sleator and R.E. Tarjan, "Amortized efficiency of list update and paging rules", Comm. Assoc. for Comp. Mach., vol. 28, no. 2, Feb. 1985.

[20] J.S. Vitter, "Design and analysis of dynamic Huffman coding", Proceedings of the 26th Annual Symposium on Foundations of Computer Science, pp. 293-302, IEEE Computer Society Press, October 1985.

1985. [21] T.A. Welch, "A technique for high-performance data compression", IEEE Computer, vol. 17, no. 6, pp. 8-19, June 1984.

[22] A.D. Wyner, "An upper bound on the entropy series", Inform. Contr., vol. 20, pp.176-181, 1972.

[23] J. Ziv and A. Lempel, "A universal algorithm for data compression", IEEE Trans. Inform. Theory, vol. IT-23, no. 3, pp. 337-343, May 1977.

[24] J. Ziv, "Coding theorems for individual sequences", IEEE Trans. Inform. Theory, vol. IT-24, no. 4, pp. 405-412, July 1978.

[25] J. Ziv and A. Lempel, "Compression of indivdual sequences via variable-rate coding", IEEE Trans. Inform. Theory, vol. IT-24, no.5, pp. 530-536, September 1978.