6. REMARKS

We have described a simple data compression scheme and analyzed its performance both theoretically and experimentally. Both analyses suggest that the method may be useful in practice. An intriguing area for future research is to devise other locally adaptive data compression schemes and compare them with the move-to-front scheme. Dynamic Huffman coding can be made locally adaptive by keeping a "window" as suggested by Knuth [16], maintaining a Huffman tree for word frequencies within the window. Another possibility is to maintain a dynamic Huffman tree based on a weight for each word that is incremented by one each time the word is compressed; periodically all word weights are multiplied by a constant factor less than one. Recently Elias [8] independently discovered the move-to-front scheme and derived inequalities (1) and (2). He also proposed a related scheme called *interval coding*, in which a word is encoded as a prefix code of the number of words occuring since its last appearance. Elias showed that inequalities (1) and (2) hold for interval coding (which also follows from our analysis). Interval coding always needs at least as many bits as the move-to-front scheme but is easier to implement. It would be useful to derive further results comparing these locally adaptive schemes.

It is important to note that with our scheme loss of synchronization between sender and receiver can be catastrophic, whereas this is not true with static Huffman coding. This suggests the study of adaptive schemes that might overcome this problem.

APPENDIX: ANALYSIS

To analyze our scheme we need to have specific prefix codes for the integers. Elias [7] and Bentley and Yao [3] describe a series of encoding schemes in which the integer i is encoded with roughly $\log i$ bits. The various schemes differ in their choice of trade-off between performance on small numbers and performance on large numbers.

The simplest of the schemes encodes the integer $i \geq 1$ with $1 + 2\lfloor \log i \rfloor$ bits. The encoding of i consists of $\lfloor \log i \rfloor$ 0's followed by the binary representation of i (which takes $1 + \lfloor \log i \rfloor$ bits, the first of which is a 1). This results in a prefix code since the total length of a codeword is exactly one plus twice the number of 0's in the prefix. Once the length is known the boundary between codewords can be found.

Another scheme results if we replace the $\lfloor \log i \rfloor$ 0's followed by a 1, by a two part prefix (an encoding of $1 + \lfloor \log i \rfloor$ by the above scheme) which takes $1 + 2\lfloor \log(1 + \lfloor \log i \rfloor) \rfloor$ bits. Thus we have a scheme that encodes i with $1 + \lfloor \log i \rfloor + 2\lfloor \log(1 + \log i) \rfloor$ bits. (Note that $\lfloor \log(1 + \lfloor \log i \rfloor) \rfloor = \lfloor \log(1 + \log i) \rfloor$.)

These ideas can be applied again to give an encoding for i with $1 + \lfloor \log i \rfloor + \lfloor \log(1 + \log i) \rfloor + 2\lfloor \log(1 + \log(1 + \log i)) \rfloor$ bits. This process can be continued; however, the codes that result are better only for astronomically large numbers.

Knowing the range of numbers to be encoded in advance can be used to advantage. For example, if the numbers are bounded above by n, then in the first scheme the $\lfloor \log i \rfloor$ 0's followed by a 1 can be replaced by $\lfloor \log(1 + \log n) \rfloor$ bits, giving an encoding for i with $\lfloor \log i \rfloor + \lfloor \log(1 + \log n) \rfloor$ bits. The same idea applied to the second scheme gives an encoding of i in $\lfloor \log i \rfloor + \lfloor \log(1 + \log i) \rfloor + \lfloor \log(1 + \log(1 + \log n)) \rfloor$ bits.

For the following discussion we assume that an encoding of the integers has been chosen, and that the number of bits needed to encode the integer i is at most $f(i)$, where $f(i)$ is a concave monotonically increasing function defined on real values of $i \geq 1$. For example, if we choose the second scheme then we can let $f(i) = 1 + \log i + 2 \log(1 + \log i)$. We assume that the input stream has been partitioned into a sequence of dictionary words, which we shall call symbols. Let the sequence of symbols be $X = x_1, x_2, \ldots, x_N$. The symbols are taken from a dictionary S of size n. Let $\rho_{MF}(X, f)$ be the average number of bits per symbol needed to transmit X by the move-to-front scheme using a code with codeword length function f. That is, $\rho_{MF}(X)$ is the total number of bits needed to transmit the sequence X divided by N. (From now on we omit the reference to f.) Let N_a be the number of occurrences of a symbol a in X. Then we have

THEOREM 1.

$$\rho_{MF}(X) \leq \sum_{a \in S} \frac{N_a}{N} f\left(\frac{N}{N_a}\right). \tag{6}$$

PROOF.

Let $t_1, t_2, \ldots, t_{N_a}$ be the times when the N_a occurrences of the symbol a are sent. That is, $x_{t_i} = a$ and $t_i < t_{i+1}$. When a occurs at time t_1 its position in the list is at most t_1. Furthermore, when a occurs at time t_i for $i > 1$ its position is at most $t_i - t_{i-1}$. Therefore the cost of transmitting the first a is at most $f(t_1)$, and the cost of transmitting the ith a is at most $f(t_i - t_{i-1})$. If we let $R_a(X)$ be the total number of bits used to transmit the N_a occurrences of symbol a then

$$R_a(X) \leq f(t_1) + \sum_{i=2}^{N_a} f(t_i - t_{i-1}). \tag{7}$$

Noting the concavity of f and applying Jensen's

inequality[2] we get

$$R_a(X) \le N_a f\left(\frac{1}{N_a}\left(t_1 + \sum_{i=2}^{N_a}(t_i - t_{i-1})\right)\right) = N_a f\left(\frac{t_{N_a}}{N_a}\right) \le N_a f\left(\frac{N}{N_a}\right). \quad (8)$$

The equality follows from the fact that the terms $t_i - t_{i-1}$ telescope, and the second inequality follows from the fact that f is monotonically increasing. Summing over all $a \in S$ and dividing by N gives Theorem 1. □

By combining Theorem 1 with a particular encoding scheme we can relate the efficiency of the move-to-front compression scheme on a particular sequence to the value of the "empirical entropy" of the sequence. This entropy, H^*, is defined as follows.

$$H^*(X) \equiv \sum_{a \in S} -\frac{N_a}{N} \log \frac{N_a}{N} \quad (9)$$

COROLLARY 1.

$$\rho_{MF}(X) \le 1 + H^*(X) + 2 \log(1 + H^*(X)). \quad (10)$$

PROOF.
We use the function f appropriate for the second scheme: $f(i) = 1 + \log i + 2 \log(1 + \log i)$.

Substituting this into Theorem 1 we get:

$$\rho_{Mf}(X) \le \sum_{a \in S} \frac{N_a}{N} + \sum_{a \in S} \frac{N_a}{N} \log \frac{N}{N_a} + \sum_{a \in S} \frac{N_a}{N} 2 \log\left(1 + \log \frac{N}{N_a}\right) \quad (11)$$

The value of the first sum is 1. The second sum is just $H^*(X)$.

Because log is a concave function and $\Sigma (N_a/N) = 1$ we can apply Jensen's inequality to the third summation to bound it by

$$2 \log\left[\sum_{a \in S} \frac{N_a}{N}\left(1 + \log \frac{N}{N_a}\right)\right]. \quad (12)$$

The summation in (12) is just $1 + H^*(X)$. Combining these results yields the corollary. □

We may now compare the performance of the move-to-front scheme with that of an optimum static prefix code for any particular sequence. One way of getting an optimum code for a particular sequence is to generate an optimum code for a source in which the probability of a symbol a occurring is N_a/N, which is just a Huffman code for this probability distribution. Let $\rho_H(X)$ be the average number of bits per symbol used by this code on the sequence X. A well known fact about an optimum static code is

$$H^*(X) \le \rho_H(X) \le H^*(X) + 1. \quad (13)$$

(See Gallager [9, ch. 3].)

Substituting the left hand inequality into Corollary 1 gives us inequality (2) in Section 3, namely

$$\rho_{MF}(X) \le 1 + \rho_H(X) + 2 \log(1 + \rho_H(X)).$$

This means that the move-to-front scheme at its worst performs almost as well as a static optimum code, even though it has no advance knowledge of the sequence. Moreover, the move-to-front scheme will do much better than the static optimum on certain types of sequences.

We can also evaluate the average performance of our scheme when compressing a sequence of symbols generated independently according to a fixed distribution. (This is called a discrete memoryless source.)

THEOREM 2.
If the symbols are generated by a discrete memoryless source in which Prob$\{x_1 = a\} = P_a$, then we have

$$\langle \rho_{MF}(X) \rangle \le \sum_{a \in S} P_a f\left(\frac{1}{P_a}\right), \quad (14)$$

where $\langle \cdot \rangle$ denotes expected value over all sequences of length N.

PROOF.
Taking expected values on both sides of Theorem 1, we have

$$\begin{aligned} &\langle \rho_{MF}(X) \rangle \\ &\le \sum_{a \in S} \left\langle \frac{N_a}{N} f\left(\frac{N}{N_a}\right) \right\rangle \\ &= \sum_{a \in S} \sum_{i=1}^{N} \left(\text{Prob}\{N_a = i\} \frac{i}{N} f\left(\frac{N}{i}\right)\right) \\ &= \sum_{a \in S} \sum_{i=1}^{N} \left(\binom{N}{i} P_a^i (1 - P_a)^{N-i} \frac{i}{N} f\left(\frac{N}{i}\right)\right) \\ &= \sum_{a \in S} P_a \sum_{i=1}^{N} \left(\binom{N}{i} P_a^{i-1} (1 - P_a)^{N-i} \frac{i}{N} f\left(\frac{N}{i}\right)\right). \end{aligned} \quad (15)$$

The next step is to pull f out of the inner summation using Jensen's inequality. To do this we must verify that

$$\sum_{i=1}^{N} \binom{N}{i} P_a^{i-1} (1 - P_a)^{N-1} \frac{i}{N} = 1.$$

This follows immediately from the observation that

$$\binom{N}{i} \frac{i}{N} = \binom{N-1}{i-1}$$

and the binomial theorem.

[2] Jensen's inequality states that if f is a concave function, $\{w_i\}$ is a set of positive real weights whose sum is 1, and $\{p_i\}$ is a set of points in the domain of f, then $\sum_i w_i f(p_i) \le f(\sum_i w_i p_i)$ [11].