

Move to front

1. (a) *Decode* the following sequence of words. Assume the sequence was encoded using the move-to-front algorithm. C is a code for position and C_{raw} is a code for raw words.

$C(1), C_{raw}(\text{LIKE}), C(2), C_{raw}(\text{I}), C(3), C_{raw}(\text{KNOW}), C(4), C_{raw}(\text{YOU}), C(4), C(5),$
 $C_{raw}(\text{COFFEE}), C(2), C(3), C(4), C(5), C(4), C(5)$

- (b) Describe a prefix code that could be used for C_{raw} . Assume the file to be encoded is an ASCII file consisting of a sequence of words, separated by blanks which you don't encode (as in (a)).

2. Consider the string

THE CAR ON THE LEFT HIT THE CAR I LEFT

Parse the string into words. Run the move-to-front algorithm on this sequence of words. Assume that:

- words are encoded using a raw code C_{raw} e.g. a sequence of ASCII chars for the letters, ended by a NULL char (ASCII 0);
- positions of words in the list are encoded using code $C(i)$

3. Suppose that a sequence $X_1 \dots X_n$ of random variables is such X_j and X_{j+1} never have the same value. That is,

$$p(X_{j+1} = A_i \mid X_j = A_i) = 0$$

for all j and for all $i = 1, \dots, N$ where N is the number of symbols in the alphabet.

- (a) How could a move-to-front code be chosen to exploit this property?
- (b) In addition to the above property, suppose that there is a constant p_0 such that for all $i \neq k$,

$$p(X_{j+1} = A_i \mid X_j = A_k) = p_0 .$$

- What is this constant p_0 ?
 - Suggest a code on positions in the move-to-front list that would be appropriate in this case.
4. Suppose we were to encode an *inverted file* using the Elias code. i.e. For each symbol A_i , we Elias-encode the inverted list, namely the first occurrence A_i and the gap sizes between successive occurrences. (We would also need to encode the number of occurrences n_i for each A_i but we ignore that here.)

Give a worst case upper bound on the number of bits used for each Elias-encoded inverted list. Do *not* make any assumptions about the frequency counts n_i of the symbols.

Hint: recall the derivation in class of the worst-case upper bound on move-to-front.

Lempel Ziv

1. Encode the characters in the string

RALF#ATE#ALFALFA

using:

- (a) LZ2 (sliding window, where the window is large)
- (b) LZ3 (match to longest previous phrase)

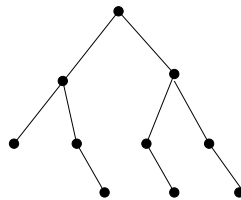
For each, give the parsing of the string into phrases, and specify what needs to be coded for each phrase. In addition, for LZ3, give the table that would be constructed by the decoder.

2. How could Huffman coding be used within LZ2 (sliding window) ?
3. (a) Suppose the following (length, symbol/offset) pairs are encoded by the Lempel-Ziv sliding window algorithm (LZ2):

(0, a), (0, b), (1, 2), (2, 3), (2, 4), (3, 2)

Decode to find the original sequence of symbols.

- (b) Suppose LZ3 encodes a particular sequence of symbols from alphabet {a, b} such that the encoder tree has the form:



Assuming that each left child is a and each right child is b, give an example of a sequence of symbols that could have produced this encoder tree.

- (c) Suppose LZ3 encodes a particular sequence from alphabet {a, b, c} as follows:

(0, a), (0, b), (1, c), (0, c), (4, a), (3, a)

Decode to find the sequence.

4. (a) Parse the following string using Lempel-Ziv 2 (sliding window)

0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1

namely five consecutive occurrences of 0 0 0 0 1. Assume the window size is longer than the string.

- (b) In general, for Lempel-Ziv 2, one may be able to use fewer bits if one encodes only longer phrases using the offset, but encodes shorter phrases using a raw symbol code i.e. based on $C(A_i)$.

Give a rule for how long a phrase must be for the offset code to use fewer bits than a raw symbol code.

Assume the raw symbol code $C(A_i)$ is a fixed length code and assume that offsets in the sliding window are equally likely. Also assume that the decoder knows the rule!

- (c) Parse the string in (a) using Lempel-Ziv 3 (best match to previous phrase) and draw the phrase tree. Label the nodes by phrase number.

Markov Models

1. Consider a sequence $X_1 X_2 \dots X_n$ of random variables whose events are drawn from an alphabet $\{A_1, A_2, A_3\}$. Assume the conditional probabilities obey a first order Markov model such that, for all i ,

$$p(X_{i+1} = A_j \mid X_i = A_k) = \frac{3}{4} \quad \text{when } j = k.$$

and

$$p(X_{i+1} = A_j \mid X_i = A_k) = \frac{1}{8} \quad \text{when } j \neq k.$$

- (a) What are the values of the joint probability function $p(X_{i+1}, X_i)$, assuming (temporarily) that the sequence is stationary. Your answer should be a 3×3 matrix.
- (b) Show that if the first symbol in the sequence has probabilities:

$$p(X_1 = A_1) = \frac{1}{4}, \quad p(X_1 = A_2) = \frac{1}{4}, \quad p(X_1 = A_3) = \frac{1}{2}.$$

then the sequence of random variables is *not* stationary.

- (c) For the non-stationary situation in (b), calculate the probability of the particular sequence:

$$\vec{x} = A_3 A_1 A_1 A_3 A_1$$

2. Consider a sequence of random variables X_1, X_2, \dots, X_n whose values are drawn from an alphabet $\{A_1, A_2, A_3\}$.

Suppose the sequence is partitioned into consecutive pairs,

$$(X_1, X_2) (X_3, X_4) (X_5, X_6) \dots (X_{n-1}, X_n)$$

where n is even.

Let the *joint* probability function $p(X_{2j}, X_{2j-1})$ be represented by the matrix:

$$P = \begin{bmatrix} \frac{1}{24} & \frac{4}{24} & \frac{2}{24} \\ \frac{4}{24} & \frac{2}{24} & \frac{2}{24} \\ \frac{3}{24} & \frac{2}{24} & \frac{4}{24} \end{bmatrix}$$

whose row i and column k hold the joint probability

$$p(X_{2j} = A_i, X_{2j-1} = A_k).$$

- (a) What are the *marginal* probability functions $p(X_{2j})$ and $p(X_{2j-1})$?
- (b) Is the sequence $X_1X_2 \dots X_n$ stationary? Explain.
- (c) Give a 3×3 matrix representing the *conditional* probabilities $p(X_{2j} | X_{2j-1})$.
- (d) Give an expression for the entropy of the pair (X_{2j}, X_{2j-1}) .

Header files

1. Suppose an encoder wished to compress an ASCII file by parsing the file into disjoint phrases, and encoding each phrase. Let each phrase be defined as the longest sequence of ASCII characters from one and only one of the following pre-defined groups:
 - letters (such as `a A b B ...`)
 - digits, logical and arithmetic symbols (such as `0 1 2 & * / + ...`)
 - control characters (such as `ACK NULL ...`)
 - punctuation and other symbols (such as `. , ; : ? ‘ [] blank ...`)
- (a) Briefly describe a code for such phrases. The code should be based both on the ASCII code and on the four groups. The code defines one codeword per phrase.
- (b) How could the encoder/decoder improve the code in (a) by taking account of the probabilities of occurrences of the groups and of each group element? Assume that the encoder/decoder can agree in advance on the probabilities and on the code.
- (c) Suppose that the encoder wished to choose an optimal code for phrases that was based on the occurrence frequencies for a particular file. How could the encoder specify this code as a header file? This is just an extension of (b) where the encoder first needs to communicate to the decoder what the code is.
Assume that the encoder/decoder have agreed in advance what are the elements in each of the four groups.