UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

B. Sc. Examination 2003

COMPUTING AND INFORMATION SYSTEMS

IS52010A(CIS325) Data Compression

Duration: 2 hours and 15 minutes

Date and time:

Answer FOUR questions.
Full marks will be awarded for complete answers to FOUR questions.
There are 100 marks available on this paper
Electronic calculators may be used. The make and model should be specified on the script and the calculator must not be programmed prior to the examination.

THIS EXAMINATION PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM

Question 1 (a) Explain the meaning of the following terms: [8]

(i) rate-distortion problem

(ii) entropy

(iii) prefix codes

(iv) motion prediction.

(b) Explain the concept of the following and provide one example for each case. [8]

(i) fixed-to-variable model

(ii) gray-scale image

(c) It has been suggested that unique decodability is not a problem for any fixed length codeword set. Explain why this is so with the aid of an example. [4]

(d) Provide reasons, with the aid of an example, to support the following statement: [5]

> "The Huffman code in general is not optimal unless all probabilities are negative powers of 2."

Question 2 (a) Comment on the *truth* of the following statement in describing the absolute limits on lossless compression. [4]

"No algorithm can compress even 1% of all files, even by one byte."

(b) Explain briefly what is meant by *Canonical* and *Minimum-variance* Huffman coding, and why it is possible to derive two different Huffman codes for the same probability distribution of an alphabet. [4]

(c) Describe briefly, with the aid of an example, how Shannon-Fano encoding differs from *static* Huffman coding. [5]

Hint: You may give an example by deriving a code for {A,B,C,D} with probability distribution 0.5, 0.3, 0.1, 0.1 using the two algorithms, and show the difference.

(d) A binary tree (0-1 tree) can be used to represent a code containing a few codewords of variable length. Consider each of the four codes for characters A,B,C,D below and draw the binary trees for each code.

(i) {000, 001, 110, 111}

(ii) {110,111,0,1}

(iii) {0000,0001,1,001}

(iv) {0001,0000,0001,1}

For each tree drawn, comment on whether the code being represented by the binary tree is a prefix code and give your reasons. [12]

Question 3 (a) Given an alphabet of four symbols {A,B,C,D}, discuss the possibility of a uniquely decodable code in which the codeword for A has length 1, that for B has length 2 and for both C and D have length 3. Give your reasons. [4]

(b) What would be the output if the HDC algorithm is applied to the sequence below? Explain the meaning of each control symbol that you use. [5]

TTU␣␣␣␣␣␣␣␣␣K␣␣RR33333333333333␣␣PPPEE

(c) One way to improve the efficiency of Huffman coding is to maintain two sorted lists during the encoding process. Using this approach, derive a canonical minimum variance Huffman code for the alphabet {A,B,C,D,E,F} with the probabilities (in %) 34,25,13,12,9,7 respectively. [8]

(d) Explain the concept of bitmapped images and vector graphics. What are the differences between vector and bitmapped graphics in terms of *requirements* to the computer storage capacity, and the *size* of the image files. [8]

Question 4 (a) Determine whether the following codes for the alphabet {A, B, C, D} are *uniquely decodable*. Give your reasons for each case. [8]

(i) {1, 10, 101, 0101}

(ii) {000, 001, 010, 111}

(iii) {0, 001, 10, 011}

(iv) {000, 010, 011, 1}

(b) Consider a source with a binary alphabet {B, W}. Under what condition on the source does the static Huffman coding algorithm achieve the best performance? Under what condition on the source does the algorithm perform the worst? Give your reasons. [7]

(c) Illustrate how adaptive Huffman encoding works. [6]

Hint: You may demonstrate step by step the progress of running an Adaptive Huffman encoding algorithm in terms of the input, output, alphabet *and the* tree structure. *Suppose the sequence of symbols to be encoded from the initial state is* CAAABB.

(d) Explain what is used to represent the so-called colour depth in a common RGB colour model. What is the value of the colour depth in a representation where 8 bits is assigned to every pixel? [4]

Question 5 (a) Huffman coding can perform badly when it is applied to fax images. For example, the canonical minimum-variance Huffman code is about 37% worse than its optimal when the probabilities of the white pixels $Pr(White) = 0.2$ and $Pr(Black) = 0.8$. Demonstrate step by step how this situation can be improved by blocking two symbols at a time from the source. [12]

Hint: $\log_{10} 2 \approx 0.3$; $\log_{10} 0.8 \approx -0.1$; $\log_{10} 0.2 \approx -0.7$.

(b) Derive the *reflected Gray code* for decimal number 5. [3]

(c) A binary sequence of length 5 (symbols) was encoded on the binary alphabet {A,B} using Arithmetic coding. Suppose the probability $Pr(A) = 0.2$. If the encoded output is 0.24, derive the decoded output step by step. [10]

The decoding algorithm is given below.

```
1. (Initialise) L <- 0 and d <- 1
2. (Probabilities) For the next symbol,
     Let Pr[s1] be p1, the probability for s1
     and Pr[s2] be p2, the probability for s2
3. (Update) If x is a member of [L,L+d*p1)
     then output s1, leave L unchanged, and
          set d<-d*p1
     else if x is a member of [L+d*p1, L+d)
          then output s2, set L<- L+d*p1 and d<-d*p2
4. (Done?) If the_number_of_decoded_symbols
              < the_required_number_of_symbols
     then go to step 2.
```

Question 6 (a) Consider part of a grayscale image with 16 shades of gray that is represented by the array A below:

```
0011 1000 1000 0010
1100 1000 1100 0110
1000 1100 1001 1001
```

Demonstrate how the image can be represented by several bitplanes (bi-level images) [4]

(b) Describe the main idea of *predictive encoding*. Suppose the matrix below represents the pixel values (in decimal) of part of a grayscale image. Let the prediction be that each pixel is the same as the one to its left. Illustrate step by step how predictive encoding may be applied to it. [6]

```
1 1 1 1
5 1 1 1
5 5 5 5
7 9 4 5
```

(c) Demonstrate step by step how the Basic LZW *encoding* and *decoding* algorithms maintain the same version of a dictionary without ever transmitting it in a separate file, using a small string `AABABC` as an example. [15]

The LZW encoding and decoding algorithms are given below.

(i) Encoding

```
word='';
while not end of file
  x=read next character;
  if word+x is in the dictionary
    word=word+x;
  else
    send the dictionary number for word (to output)
    add word+x to the dictionary
    word=x
end of while loop
output the number for word
```

(ii) Decoding

```
read a character x from compressed file;
write x to uncompressed version;
word=x's uncompressed version;
while not end of the compressed file do
  begin
    read x
    look up dictionary element corresponding to x;
    output element
    add word+first char of element to the dictionary
    word=element
  endwhile
```