# Homework 1: Lossless Coding

In this homework you are required to implement the lossless coding schemes to compress the following two images: **lena** and **baboon**. They are both gray-level 256x256 raw images and each sample is of 1 byte long. (You may download irfanview from http://www.irfanview.com (plus the plug-in) to view the image.). Each image has three versions: gray-level images, binary images and halftone images.

1. Huffman coding

   Use both the original data byte and DPCM as the data samples and then implement your Huffman coder to compress the image files. Find a way to transmit the codewords in the bitstream. Report the compressed file sizes (excluding the size of transmitting codewords) and compare the results with the first-order entropy. By the way, you may have problems at the end of the decoding process as you may decode more than 256x256 pixels. Let’s just ignore the problem as we assume the decoder knows the image resolution.

2. Adaptive Huffman coding

   (Bonus) Implement the adaptive Huffman coding according to the algorithm shown in the textbook. Report the compressed file sizes and compare the results with the zero-order entropy by using the original data byte and DPCM as the symbol set.

3. Binary arithmetic coding, QM encoder

   Please implement the QM **encoder.** There are two parameters A and C in the coder, where A records the size of the tag interval with a value between 0.75 (0x8000) and 1.5 (0x10000) and C is the lower bound (that stores the encoded bits.) Below is the pseudo-code of a binary arithmetic **encoder**.

```
Initialization:
Less Probable Symbol (LSP)='1'
More Probable Symbol (MPS)='0'.
Prob(LPS)=Qc=0.49582. State=0. A=0x10000. C=0x0000.
if encoder receives MPS
```

```
{
  A=A-Qc;
  if A<0x8000
  {
    if A<Qc
    {C+=A; A=Qc;}
    A<<=1;
    Qc changes its state according to Column 4 in Table 1;
    /* EX: Qc=01FB(state29) and changes to 01A4(state30)*
    /* because column 4 indicates increasing 1*/
    encoder outputs MSB of C;
    C<<=1;
  }
}
if encoder receives LPS
{
  A=A-Qc;
  if A>=Qc
  {C+=A; A=Qc;}
  A<<=1;
  Qc changes its state according to Column 5 in Table 1;
  /* EX: Qc=32B4(state08) and changes to 3C3D(state06)*/
  /* because column 5 indicates decreasing 2*/
  encoder outputs MSB of C;
  C<<=1;
}
```

Again, you are required to compress the images. For gray-level images, in order to generate binary symbols, please process each bit-plane (from MSB to LSB, 8 bit-plane) separately and report the bit-stream length of each bit-plane. Try Gray code as well. For binary image, you can process them directly.

4. (Bonus) QM decoder

Please implement the decoder by studying the reference document. You have to modify the encoder to make the encodec work.

The deadline is **Nov. 13, 2019 23:59am**. Please submit your homework through the course website. The homework should be compressed into one file (.zip or .rar) with your student ID as the filename (ID#.zip or ID#.rar). The zipped file should contain your source code, the necessary compiling information and the **report**.

**Table 1: State Change of Binary Arithmetic Coder**

| State | Qc (Hex) | Qc (Dec) | Increase state by | Decrease state by |
|---|---|---|---|---|
| 0 | 59EB | 0.49582 | 1 | S |
| 1 | 5522 | 0.46944 | 1 | 1 |
| 2 | 504F | 0.44283 | 1 | 1 |
| 3 | 4B85 | 0.41643 | 1 | 1 |
| 4 | 4639 | 0.38722 | 1 | 1 |
| 5 | 415E | 0.36044 | 1 | 1 |
| 6 | 3C3D | 0.33216 | 1 | 1 |
| 7 | 375E | 0.30530 | 1 | 1 |
| 8 | 32B4 | 0.27958 | 1 | 2 |
| 9 | 2E17 | 0.25415 | 1 | 1 |
| 10 | 299A | 0.22940 | 1 | 2 |
| 11 | 2516 | 0.20450 | 1 | 1 |
| 12 | 1EDF | 0.17023 | 1 | 1 |
| 13 | 1AA9 | 0.14701 | 1 | 2 |
| 14 | 174E | 0.12581 | 1 | 1 |
| 15 | 1424 | 0.11106 | 1 | 2 |
| 16 | 119C | 0.09710 | 1 | 1 |
| 17 | 0F6B | 0.08502 | 1 | 2 |
| 18 | 0D51 | 0.07343 | 1 | 2 |
| 19 | 0BB6 | 0.06458 | 1 | 1 |
| 20 | 0A40 | 0.05652 | 1 | 2 |
| 21 | 0861 | 0.04620 | 1 | 2 |
| 22 | 0706 | 0.03873 | 1 | 2 |
| 23 | 05CD | 0.03199 | 1 | 2 |
| 24 | 04DE | 0.02684 | 1 | 1 |
| 25 | 040F | 0.02238 | 1 | 2 |
| 26 | 0363 | 0.01867 | 1 | 2 |
| 27 | 02D4 | 0.01559 | 1 | 2 |
| 28 | 025C | 0.01301 | 1 | 2 |
| 29 | 01F8 | 0.01086 | 1 | 2 |
| 30 | 01A4 | 0.00905 | 1 | 2 |
| 31 | 0160 | 0.00758 | 1 | 2 |
| 32 | 0125 | 0.00631 | 1 | 2 |
| 33 | 00F6 | 0.00530 | 1 | 2 |
| 34 | 00CB | 0.00437 | 1 | 2 |

| | | | | |
|---|---|---|---|---|
| 35 | 00AB | 0.00368 | 1 | 1 |
| 36 | 008F | 0.00308 | 1 | 2 |
| 37 | 0068 | 0.00224 | 1 | 2 |
| 38 | 004E | 0.00168 | 1 | 2 |
| 39 | 003B | 0.00127 | 1 | 2 |
| 40 | 002C | 0.00095 | 1 | 2 |
| 41 | 001A | 0.00056 | 1 | 3 |
| 42 | 000D | 0.00028 | 1 | 2 |
| 43 | 0006 | 0.00013 | 1 | 2 |
| 44 | 0003 | 0.00006 | 1 | 2 |
| 45 | 0001 | 0.00002 | 0 | 1 |

S means that MPS and LPS must exchange because we made a wrong guess. For example, if MPS='1', we change MPS from '1' to '0' when the encoder encounters "S."