CSCI 665 Foundations of Algorithms
Lab assignment 2
Summary Report

Srushti Kotak
sk7735

Huffman coding is an algorithm for encoding data and is also used for data compressions. A code was developed to do the same. I wrote a function which would generate random text files and encode it using huffman coding.

Another function is written which decodes the binary encoded data to the original data and writes it to a text file. I recorded the time taken to encode and decode the file. I also calculated the compression ratio and compression percentage.

Compression ratio = (Uncompressed file size)/(Compressed file size)

*Compression percentage = 100 - ((Compressed file size)/(Uncompressed file size)*100)*

---- Results in tabulated format are in the excel sheet attached- "Results_FOA_LAB_2.xlxs " ----

(Results_FOA_LAB_2.xlxs : Each entry is an average of 10 or more reruns of the execution)

---- Results in graph format are in "Results_FOA_LAB_2.pdf" -----

Observations:

1) **Compression ratio**

   (Observed for number of lines : 50, 100, 500, 1000, 2000, 5000)

   Range of compression ratios with actual frequency

   ---> 1.5910 to 1.6102 (37.15% to 37.79%)

   Range of compression ratios with estimated frequency

   ----> 1.5215 to 1.5415 (34.27% to 35.13%)

   (Observed for number of lines : 1)

   Range of compression ratios with actual frequency

   ---> 1.6823 to 1.7027 (40.90% to 41.26%)

   Range of compression ratios with estimated frequency

   ----> 1.6625 to 1.6755 (40.05% to 40.91%)

   So, based on my observation, no we cannot improve the compression ratio by using estimation of frequencies instead of actual frequencies.

2) **CPU time**

   Based on the graphical observation, the time taken with actual frequency encoding, decoding and with estimated frequency encoding, decoding is almost the same. According to me, the actual frequency encoding takes more time than estimated frequency encoding and actual frequency decoding takes less time than estimated frequency decoding. So, overall time is almost the same.

Language used : Python

Data Structure used : Tree

Instructions:

1) There are 3 sample text files attached : small, medium, large.
2) Run huffman.py.
3) Select an option to choose type of input.
   3a. Enter a filename.
   3b. Generate random text input.

Done.

References :

1. Code to generate data : http://www.bswen.com/2018/04/python-How-to-generate-random-large-file-using-python.html
2. Frequency estimates : https://en.wikipedia.org/wiki/Letter_frequency