CS201 Data Structures and Algorithms
AY2023/24 Term 1

Project Description

In this project, students work in their project groups to solve an algorithmic problem from a given image dataset involving:

• Using appropriate data structures/algorithm to solve a problem

• Running experiments and discussing the experimental outcomes

• Analyzing the trade-offs of compression/loss/time efficiencies

• Presenting the findings as a video presentation

Timeline

• Week 7: Release of project requirements (version 0.5)

• Week 8 (Recess Week): Release of project requirements (v0.9) and Source Code

• Week 9: Confirmation of project requirements

• Week 12, Friday, 6pm: Project report via 10-min video (youtube link), source codes + output file

All project clarifications should be made in Piazza.

Background

We have learnt a couple of tree structures for data representations in the course. In this project, students are required to apply **tree** structure(s) on data compression which plays a vital role in modern computing, enabling efficient data handling, transmission, and storage across various applications and industries. For example, Huffman coding is a widely used method for lossless data compression, and it involves creating a binary tree called the Huffman coding tree. The main idea behind Huffman coding is to represent more frequent characters with shorter codes and less frequent characters with longer codes. Despite the effectiveness, Huffman coding cannot achieve high compression ratios for certain types of data (e.g., multimedia data). Additionally, there are also lossy compression techniques. In fact, the tree itself can be compressed.

Requirements

Students are required to explore **tree-**based compression techniques and submit **ONE** lossy/lossless implementation. In this project, the loss threshold is set as 25%.

Students are also required to experiment with possible changes to the chosen implementation and discuss these in the **video presentation**. These changes can be regarding alternative data structures, changes/modifications to the actual algorithm, configurations or parameters etc. For each change, students are required to present the hypothesis, the change, result of the change and the analysis/trade-offs of the result. Students are encouraged to present these in the video.

Students will be able to test their implementation with a given Java Application. The implementation's final performance will be tested on a dataset containing a different set number of images using these criteria.

Grading Criteria (project will account for 20% of the overall grade for the course)

• Test Cases Quality Score (45%): performance on the test image dataset based on:

– Compressed Size (25%)

– Running Time (10%)

– Loss (10%)

• Video Presentation (40%): Explanation of your selected algorithm, clarity of presentation, experimentation, results, analyses

• X-factor (15%): What makes the project outstanding. You are welcome to justify your X-factors in the final video.

• Note that even though this is a group project, we will be conducting peer evaluation to get feedback from the respective group members that will influence the class participation components.

CS201 Data Structures and Algorithms
AY2023/24 Term 1

Source Codes (Actual Code and test images will be released in Week 8)

- In the provided source code, you are given a few Java files, most of which do not require any changes. Do understand the codes provided. The main method is in App.java.
- You can only modify Utility.java. Do not modify any other .java files or create your own .java files to ensure that we will be able to run your codes after your submission. You will only submit Utility.java
- Amend the implementation of the Compress() and Decompress() method in Utility.java.
- You are also allowed to create your own private helper methods and attributes in Utility.java should you need to
- The images will be given in a 3d int array format.
- Your Compress() function should take in this 3d array and return ONE file only. Your Decompress function would then take in this file and return a 3d array.
- If use any online resources, please state them and give credit.
- You are not allowed to import any extrnal libraries, only Java Standard libraries. ⍰
- A brief description of the key files is included in the image below.

App.java

Loop through image folder "src"
For each image:
    Convert into a 3d array
    Call Utility's Compress() function on the array
    File will be added to "compressed" folder
    Records runtime, compressed file size
    Call Utility's Decompress() function on file
    Image will be added to "decompressed" folder
    Records runtime, loss in image quality

//Functions used for conversion, runtime computation, file size calculation, loss calculation will be provided

//You are not allowed to add extra classes, and you will only submit Utility.java

Utility.java

Compress(takes in a 3d array of pixels)
// Static Method
// Student to fill this up

// Should return your compressed file

Decompress(takes in your compressed file)
// Static Method
// Student to fill this up

// Should return a 3d array of pixels

CS201 Data Structures and Algorithms
AY2023/24 Term 1

Deliverables

Upload all deliverables by **Week 12**, **Friday**, 3rd November, 6pm noon.

<u>Youtube</u>

Project report via video (max 10 min). Begin with the group name and members. Describe the problem. Describe the implementation and the key analysis of experimental results. Discuss the learning highlights, justify the x factors (what makes it special) and close with potential extensions. Please record your video at normal playback speed.

Upload the video to YouTube as an unlisted link, and provide the link via https://forms.office.com/r/cXzU871Mr9

<u>eLearn</u>

(A) Project Source Codes

(i) Submit Utility.java for the source code

Multiple submissions are allowed but only the most recent submission will be kept in the system as the final submission.