ALGORITHMS FOR OPTIMIZATION AND INFERENCE – 2024

First Assignment

Due: Monday October 28th at 23.59

Assignment Objective: *In this assignment you will be asked to design an IP model and implement an algorithm for compressing a picture by reducing the number of its colors.*

The assignment is divided in 2 parts. Part I deals with implementing the k-means algorithm seen in class. Part II deals with writing an IP model and solve it with Glpk.

Recall the definition of the k-means problem:

Given p vectors $x^i \in \mathbb{R}^n$ (for $i = 1, \dots, p$), and a number k, we wish to find k centers $C = \{c^1, \dots c^k \in \mathbb{R}^n\}$ *and assign each x^i to the closest center in C, that is, to $c(i) = \arg\min_{c^j \in C} ||x^i - c^j||^2$. The objective is to minimize the total* dissimilarity $\sum_{i=1}^{p} ||x^i - c(i)||^2$.

PART I

Problem 1:

Implement the k-means algorithm seen in class, and use it to reduce the number of colors of input pictures. In particular, your task is to write a python code (call it `recolor.py`) that takes as command-line parameters

- an input image in png format;
- the path to an output image in png format;
- the target number of colors k.

Your code will take the input image, apply the k-means algorithms, and produce the output image with k colors. It will be called as follows:

```
phyton3 recolor.py {input-image} {output-image} {k}
```

For example, I might test it on the image `20col.png` that is in the assignment directory by calling:

```
phyton3 recolor.py 20col.png 20coloutput.png 8
```

Implementation recommendation: Use the PIL (Pillow) module for reading and writing image files.

You must submit: the file `recolor.py`

PART II

Problem 2:

Write an IP model for the k-means problem.

You must submit: A pdf file in which you briefly describe your model (variables and data).

Problem 3:

Find the optimal objective function value of your IP model in Problem 2 using Glpk, for the instance `20col.png`, setting $k = 8$.

Implementation hint: Most likely, enumerating all possible centers for the instance `20col.png` (image with 20 colors) would yield an IP with too many variables, for Glpk to be able to solve it. Here is a trick to reduce the number of variables (in particular, to reduce the number of possible centers to consider):

- *Run the local search algorithm developed in PART I on the instance `20col.png`, with few random initializations. Store the best objective function value found.*
- *When considering a centroid for a possible cluster, look at the total cost yield by just this cluster: if this is larger than the best objective function value above, you know that cluster would surely not be formed, and hence there is no need to introduce a variable for its centroid...*

You must submit: the .mod file, the .dat file, and the python script that you used to generate the data file.