HPC Programming Project - 2024/2025

This project has to be done during the week 45 (from 3/11 to 10/11).

1. Main objective

This project concerns **clustering** techniques. More precisely, during this project you have to implement two methods *on the GPU* to do data clustering, based on centroid:

- Classical k-means called hereafter kMeans.
- Fuzzy k-means (also known as Fuzzy C-Means or Soft k-Means) called hereafter FCM.

Cuda must be used as most as possible to achieve this project.

1.1. Metrics

To study the efficiency of the algorithms, you will use two different metrics:

- Computation times: it should be as smaller as possible.
- Value of the cost function (should be as close as possible to 0), for instance the within-cluster sum of squares `WCSS` for k-Means [1] or `J(X,C)` for FCM. *Notice that to simplify you may consider the Euclidean distance only.*

To simplify the work, classical CPU implementation is provided as reference (see references [2] and [4]). You may do some comparison with them.

1.2. Application use case

Data clustering method based on centroid (kMeans and FkMeans) can be used for different purposes, including image segmentation (or color quantization, which is more or less the same).

Assuming a clustering class name `CM`, with:

- Method `fit(X: List) -> None` that fit the model according to data contained into the vector `X`, where data is of dimension `n` (3 for color images, 1 for gray images, any for others applications).
- Method `predict(X: List) -> List` that returns the label of each value from the vector `X` (with the same dimension). Notice that the vector used for fitting can be used for prediction again.`

An example of usage is the following one:

```
import numpy as np
from PIL import Image
from kmean import CM # a k-means API... TODO!


# read the input image
image = Image.open('images/example.jpeg')          # read image
N, M = image.size                                  # get the number of columns (N)
and rows (M)
```

```
# Transforming image into a data set
X = ( np.asarray(image).reshape((N*M, 3)) )

# Creating and fitting the model
fcm = FCM(n_clusters=10)
fcm.fit(X)

# Pixel quantization
transformed_X = fcm.predict(X)

# Converting and saving image
quantized_array = ( transformed_X.astype('uint8').reshape((M, N, 3)) )
quantized_image = Image.fromarray(np.asarray(quantized_array))
quatized_image.save('images/quantized.jpg')

# Display the two images
side_by_side = Image.fromarray(
    np.hstack([
        np.array(image),
        np.array(quantized_image)
    ])
)
side_by_side
```

2. Expected work

Your work must be sent by email including the following files.

1. A Colab file containing the different algorithms in a first part, then the metrics, and some example of usage. The image should be included, or downloaded from Internet with permanent links.
2. A short report that recaps your work, and the result obtained onto different images for the different algorithm, including the metric and the computation times.

The email is expected to be received the **Sunday 10 November, before 12pm**. Any delay is not allowed. If you miss the delay, then 0 mark will be automatically given.

4. Bibliography

[1] K-Means clustering, wikipedia, https://en.wikipedia.org/wiki/K-means_clustering

[2] K-Means implementation, https://github.com/tugrulhkarabulut/K-Means-Clustering/tree/master

[3] Fuzzy C-Means, wikipedia, https://en.wikipedia.org/wiki/Fuzzy_clustering

[4] Fuzzy C-Means implementation, https://github.com/omadson/fuzzy-c-means/tree/master