Image Segmentation using K-Means Clustering

CSE 702 (Fall ‘20)

Instructor: Dr. Russ Miller

By Sneha Panicker

University at Buffalo
School of Engineering and Applied Sciences

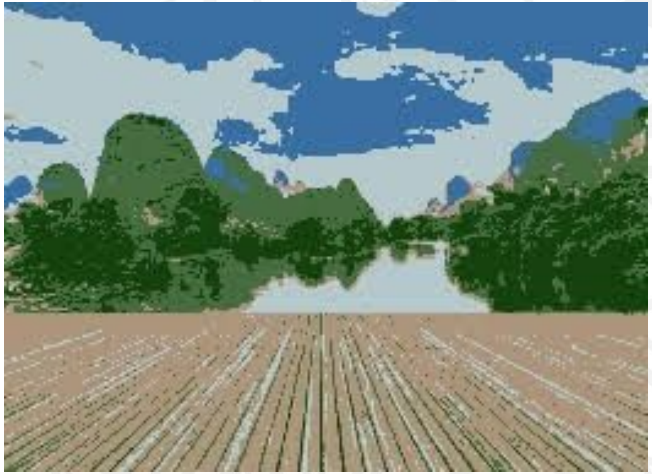
Outline

- Problem Definition
- K-Means Clustering Algorithm
- Parallel Implementation
- Results
- Time Analysis
- Observations
- References

Problem Definition

Image Segmentation using K-Means Clustering.

Original Image



Segmented Image with 5 clusters

K-Means Clustering Algorithm

1. Randomly select k pixels to be cluster centers.
2. For each pixel in the data set, associate it with the cluster which has its center closest to the pixel.
3. Calculate new cluster centers by averaging all pixels in a cluster.
4. Repeat 2 and 3 for a set number of iterations.

Parallel Implementation

Creating Dataset from Image (Serial)

1. Read the image using OpenCV for Python.
2. Append the R, G, and B values of the pixels to a list for each pixel.
3. Saving the list in a pickle file.

Parallel Implementation

1. Consider N pixels and P cores.
2. Assign N/P pixels to each core using the pickle file.
3. Core 0 randomly selects k pixels as cluster centers.
4. Each core for each of its pixels, finds the cluster to which the pixel belongs.
5. Each core calculates local sums (∑R, ∑G, ∑B) for each of the k clusters.
6. Calculate the global mean by adding all local sums from each core for each of the k clusters.
7. Repeat the clustering for the specified number of iterations. (i.e. steps 4-6)
8. Form a pickle file with information about each pixel’s final cluster center.

Segmented Image Formation (Serial)

1. Read the pickle file with information about each pixel's corresponding cluster center.
2. Read the image.
3. For each pixel, read the pixel's corresponding cluster center from our hashmap (dictionary in Python) and overwrite the pixel value with the cluster center.
4. Save the resulting image with k segments.

Results

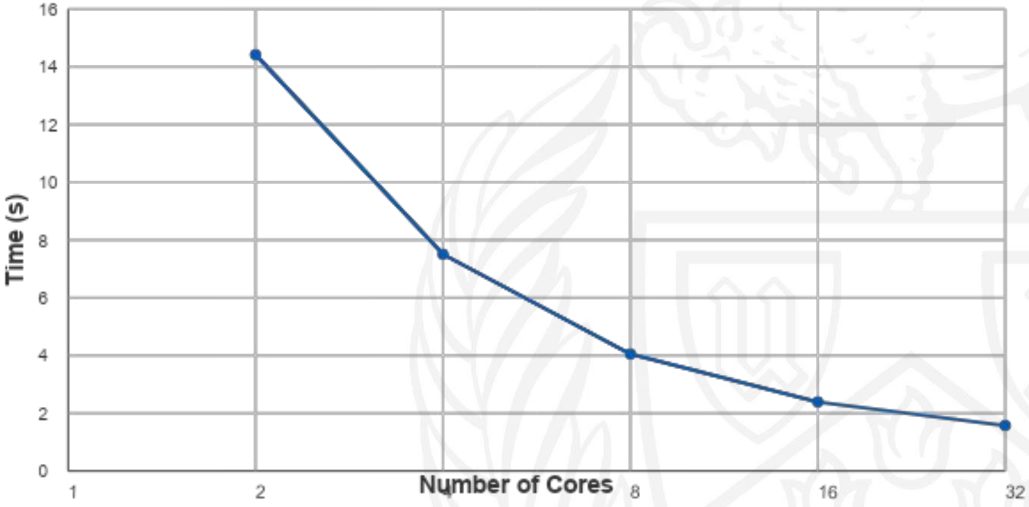
Original Image

3 Clusters

10 Clusters

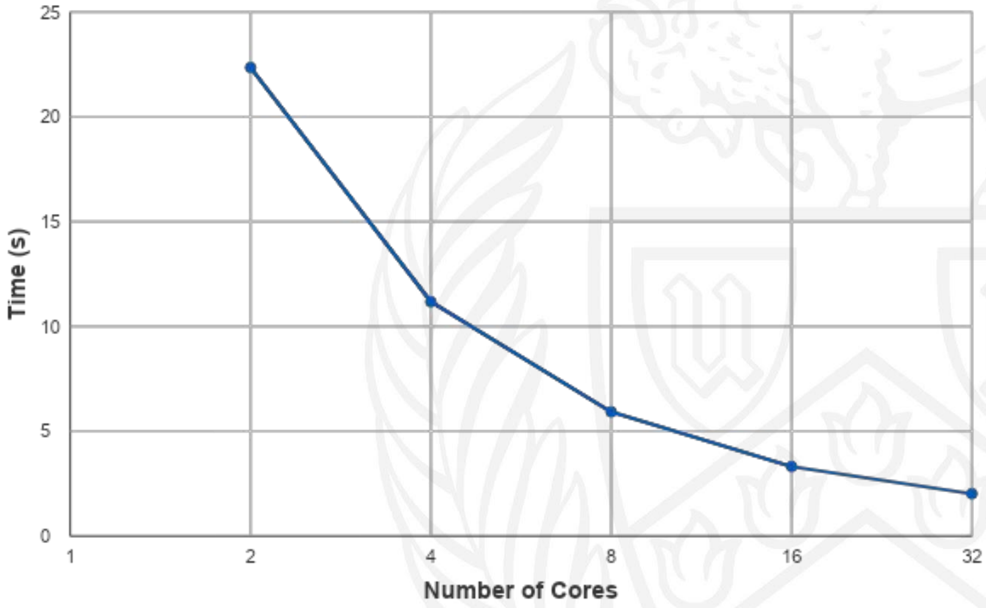
Time Analysis for 3 Clusters (256 * 256 Image)

Number of Cores	Time (s)
2	14.42
4	7.507
8	4.053
16	2.391
32	1.578



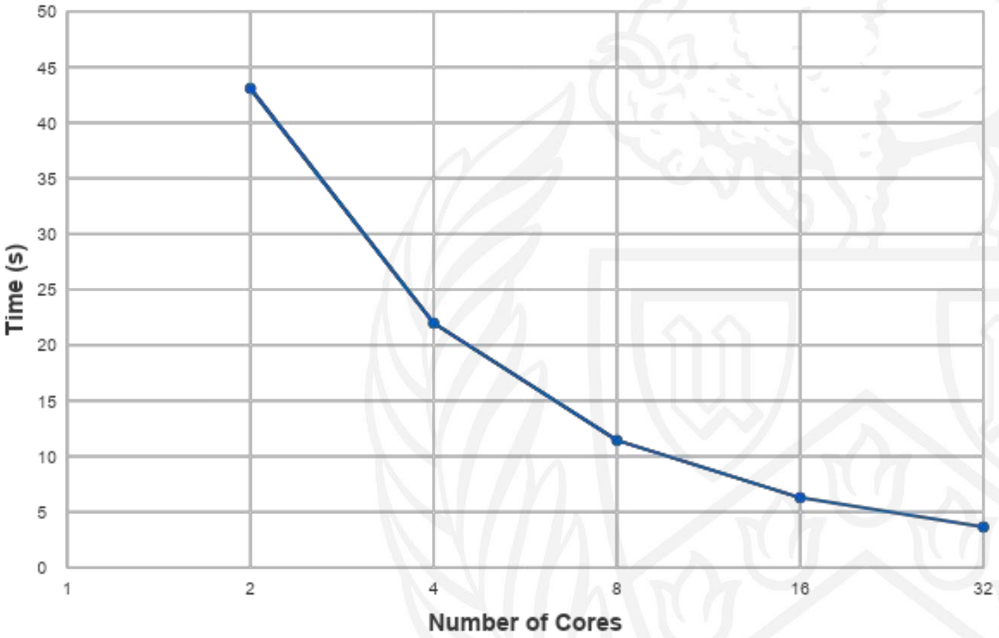
Time Analysis for 5 Clusters (256 * 256 Image)

Number of Cores	Time (s)
2	22.353
4	11.176
8	5.923
16	3.317
32	2.02



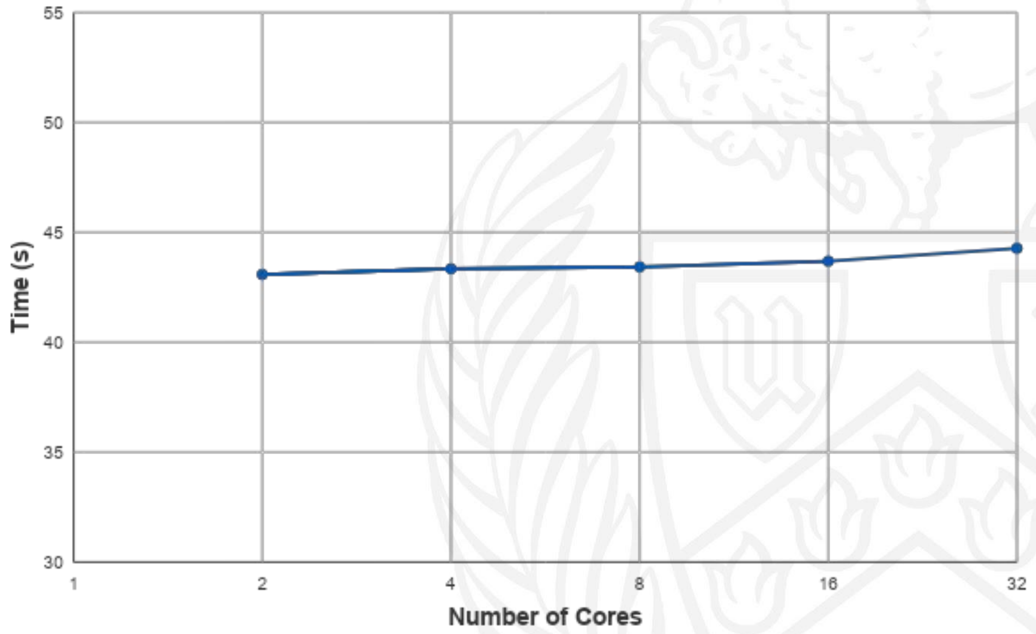
Time Analysis for 10 Clusters (256 * 256 Image)

Number of Cores	Time (s)
2	43.085
4	21.977
8	11.438
16	6.284
32	3.644



Time Analysis for Constant Data/Core

Number of Cores	Time (s)
2	43.085
4	43.343
8	43.429
16	43.688
32	44.274



Observations

- Speedup is significant when we increase the number of cores up to 8 cores for our input image.
- Using an 8 core processor is ideal when segmenting a 256*256 image since significant speedup is observed up to 8 cores.

References

- Algorithms Sequential & Parallel: A Unified Approach (Dr. Russ Miller, Dr.Laurence Boxer)
- https://towardsdatascience.com/image-compression-using-k-means-clustering-aa0c91bb0eeb
- Introduction to OpenMP

Thank You.