Optimization on K-Means Clustering with OpenMP and Comparison Study

Burak Topçu

283078027

June, 2021

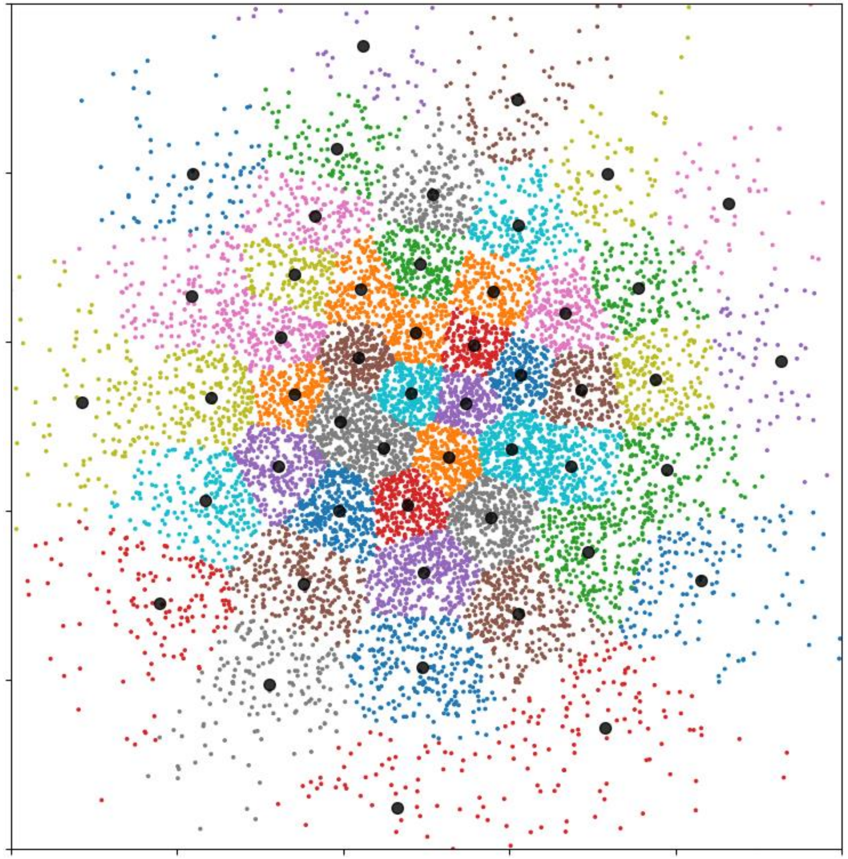
Overview

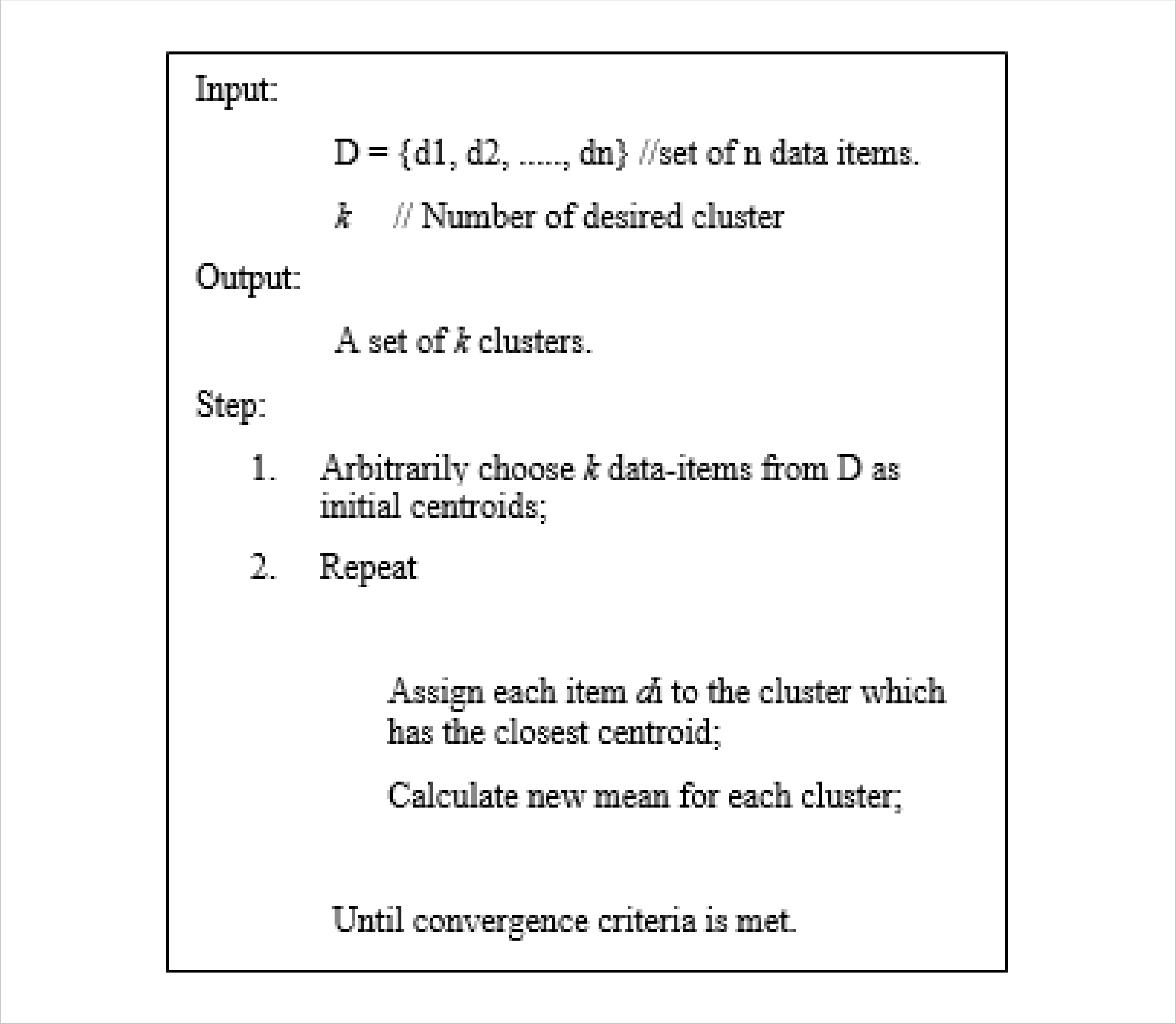
- Introduction
- K-Means Clustering
- Optimization
- Experiments
- Comparisons
- Conclusions

Introduction

- Clustering is an important concept to separate data among sets lots of industrial and scientific applications.
- One of them is the K-Means Clustering algorithm.
- Today, K-Means Clustering algorithm is being actively used in:
    - Document clustering
    - Identifying crime-prone areas
    - Customer segmentation
    - Insurance fraud detection
    - Public transport data analysis
    - Clustering of IT alerts

K-Means Clustering



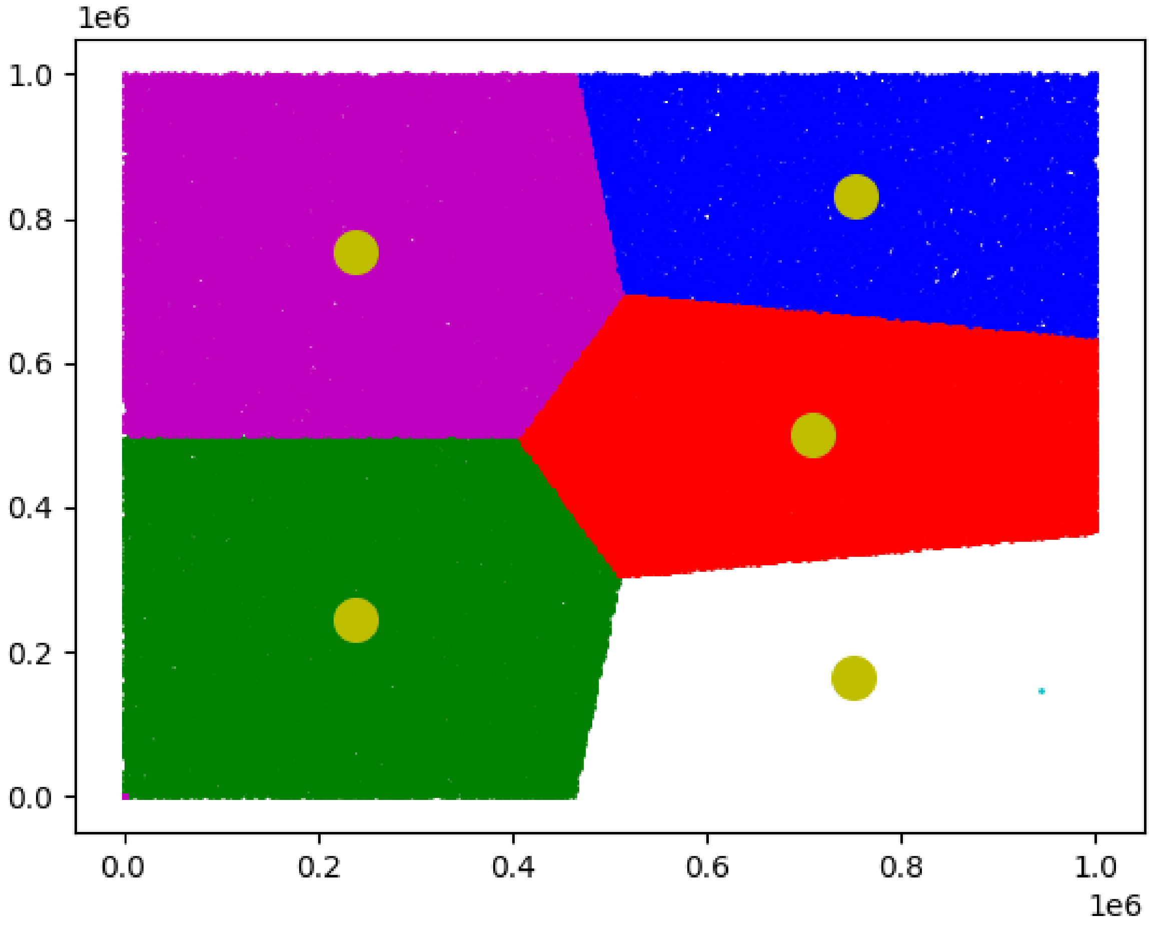


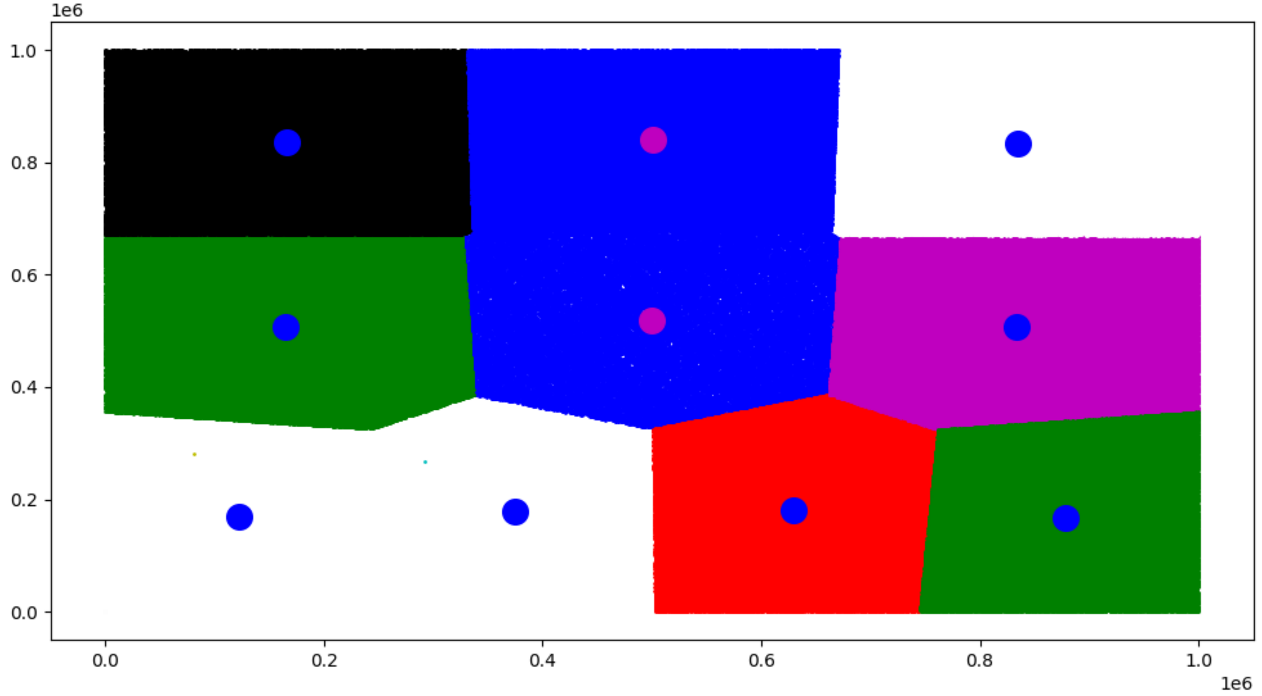
```
Input:
        D = {d1, d2, ....., dn} //set of n data items.
        k    // Number of desired cluster
Output:
        A set of k clusters.
Step:
    1.  Arbitrarily choose k data-items from D as
        initial centroids;
    2.  Repeat

            Assign each item di to the cluster which
            has the closest centroid;
            Calculate new mean for each cluster;

        Until convergence criteria is met.
```

Optimization

- There are three important parallelization pattern in K-Means Clustering algorithm:
    - Mapping while re-assigning data samples to the clusters
    - Reduction while updating the centers of the clusters by using newly assigned points
    - Fusing the reduction and mapping methods by summing the coordinates of the reassigned data samples iteratively for each corresponding cluster.

Experiments





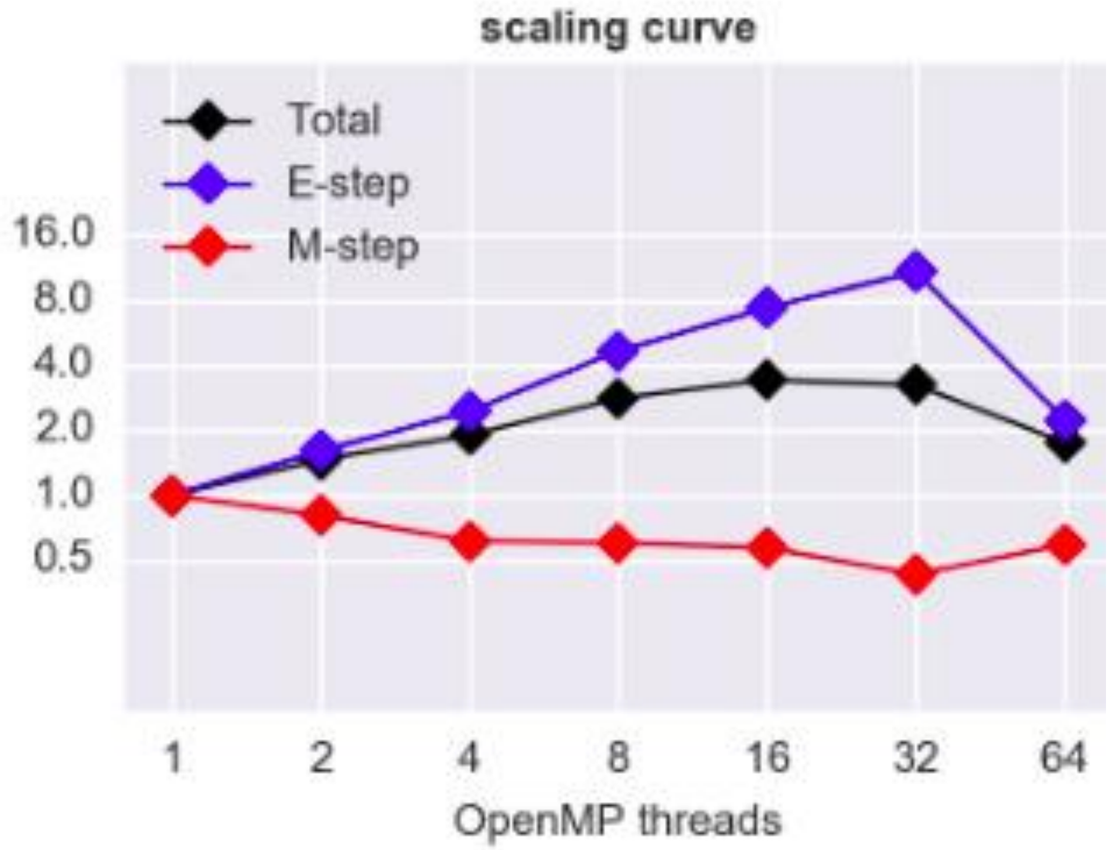
Experiments

Table 1: 1000000 samples and 10 clusters.

Applied optimization techniques	Elapsed time during the execution
With neither mapping nor reduction	424.63 seconds
With mapping and without reduction:	73.39 seconds
Without mapping and with reduction:	391.56 seconds
With both mapping and reduction:	66.44 seconds

Table 2: 250000 samples and 5 clusters.

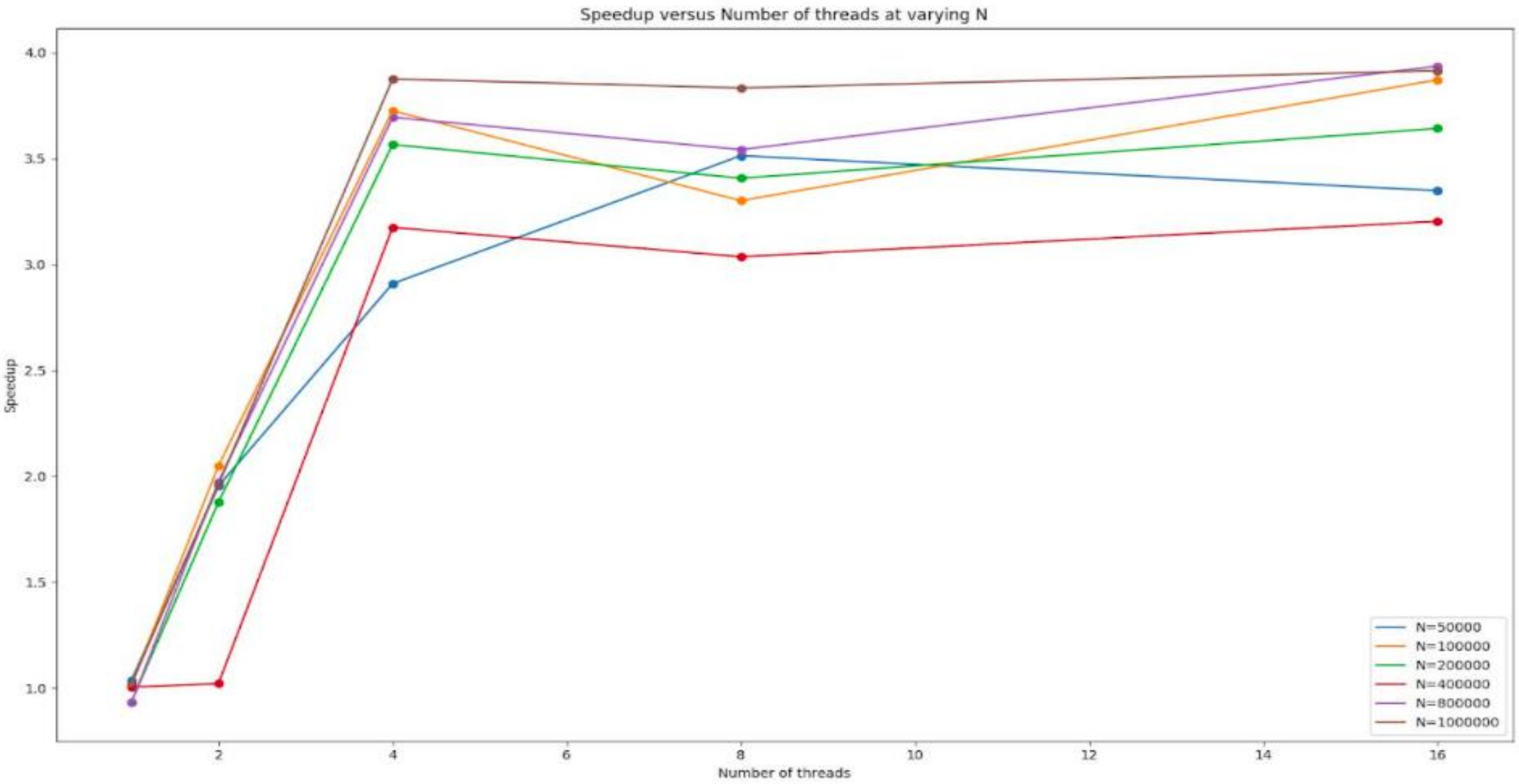
Applied optimization techniques	Elapsed time during the execution
With neither mapping nor reduction	52.88 seconds
With mapping and without reduction:	12.14 seconds
Without mapping and with reduction:	48.23 seconds
With both mapping and reduction:	10.97 seconds



Comparison

For implementaiton details: https://github.com/JiaweiZhuang/

Comparison



For implementation details: https://github.com/arneish/parallel-k-means/

Conclusions

As a result, mapping, reduction and fusing them patterns are implemented to optimize the execution of the K-Means Clustering algorithm on the belonging hardware.

This optimization is done based on the OpenMP library and implemented with C programming language.

The achieved speed up is 6.4 times faster than the non-optimized code.

Thank you all for your kind attensions.