**K-means Project - Documentation**

This paper will explain step by step the k – means parallel using MPI, OpenMP and CUDA.

**Keyword that represent MPI:** Master, Slaves, All processes and MPI.

1. **Master** read k-means data properties and points from given file name.

O(N) N -numbers of points.

1. **Master** broadcast k-means data properties to each **process (Slave).**
2. **Master** divide points array to equal size to each **process (Slave)**, and it gets the size and remainder. O(N)
3. **All processes** call kmeans function.
4. **All processes** call kmeans algorithm function.

* **Master** Chooses first K points as a cluster centers, and broadcasts cluster array to each **Slaves**. O(K)
* **All processes** Classify points to most close cluster using **CUDA**. O(N)
* **All processes** synchronize if at least one point moved to other cluster.
* **All processes** recalculate the cluster center using **OpenMP** and synchronize the new center using **MPI**. O(N)
* **All processes** check the termination condition- no moved points to other cluster or maximum iteration limit was made.
* Repeat from the second step till the termination condition fulfills.

1. **Master** gathers all the updated points from the **slaves**. O(N)
2. **Master** evaluates quality using **OpenMP**. O(N^2)
3. **Master** broadcasts to **slaves** the current quality.
4. **All processes** check if the quality less than the requested quality.
5. **All processes** check the termination condition- reached the quality or interval Time was made.
6. If the requested quality wasn't reach, time will be raised by dt, and **all processes** reposition points by the new time using **OpenMP**. O(N)
7. Repeat from the step 1 till the termination condition fulfills.

***Evaluate complexity O(N^2).***