# Assignment 3

## Psuedo Code K-Means Clustering

1. Initialize K cluster centroids randomly or K data points as cluster centroids.
2. Assign each point to the closest cluster.
3. Recalculate centroid of each cluster by taking average of coordinates of points assigned to a cluster.
4. Repeat step 1,2,3 until converge or till max no of iterations.

## Finding Number of Clusters

I have used K-Means clustering for clustering the data points. To achieve minimum euclidean distance error, number of clusters can go upto number of data points, in which case distane error will be zero. More number of clusters will definately take more time to cluster data points and will give cluster centroids with less total error. Considering the number of data points and after few experiments with different number of clusters, I found that for more than 100 clusters, some clusters getting assigned same data points i.e. they represent same cluster or some clusters are getting very few data points. After observing this, and considering time it took to create 100 clusters on HPC2010, I decided to go with 100 number of clusters and at the end merge cluster centroids having same mean or discard clusters having 0 data points assigned to them. Convergence is decided using number of data points changing their cluster id. If less than 1% of points are changing their cluster then we can assume that clustering algorithm has converged.

## Data Decomposition Strategy

Rank 0 process read data file for each timestamp and calculated no of points present.

Broadcast number_of_points in the file from root rank 0.

Scatter approximately no_of_points/P data points to each process where P is the number of processes.

Rank 0 process initializes K points as cluster centroids. K is the number of clusters we want.

Every process assignes each data point it has to the closest cluster and send that information to Rank 0 process.

After receiving information from every process, Rank 0 process calculates new cluster centroids and broadcasts them.

Discard clusters having 0 points assigned to them or having same cluster centroids.

## Some observations

1. Increasing number of processes decreases processing time.
2. After a threshold, increasing number of processes doesn’t help improving clustering time.
3. Using constant ppn=4, time for clustering sometimes increases a little bit for 5 processes compared to 4 processes and then again starts decreasing for number of processes > 5.

Note: Running sub.sh on HPC will generate 3 csv files for each dataset. While submitting I have combined 3 files into one data.csv file for each dataset. To execute on HPC run “./sub.sh”.