**Approach**

The parallelization of k-means using MPI for clustering XY-coordinate points involves four steps:

1. First, the input data is read by the root process. The reason that the data is not read in parallel on all the processes is because an MPI system makes no guarantees about the homogeneity of the file systems on the processes in the cluster.
2. Then, the root process scatters the input data across all the processes, such that each process has access only to the points on which it is personally performing calculations.
3. Then, the root initializes random means and broadcasts these means to the cluster.
4. Next, the following is repeated until convergence:

* Each process assigns their local points to clusters and calculates a sum of the points in each cluster as well as a point count for each cluster.
* Then, the processes reduce their partial sums and cluster sizes to the root, and the root updates the mean for each cluster.
* Lastly, the root broadcasts the updated means to the processes.

Pseudo code for (4):

repeat while changed:

changed <- False

for each point p in local points do:

k <- closest cluster to p

sum[k] <- sum[k] + p

count[k] <- count[k] + 1

For int k from 0 to num\_clusters – 1 do:

count\_k <- reduce to all procs count[k]

sum\_k <- reduce sum[k] to sum[k] on root

if this process is root:

mean\_k <- sum\_k / count\_k

if mean\_k != means[k]:

means[k] <- mean\_k

changed <- True

Broadcast means from root to all procs

**Experimentations and Results**

We ran our parallelized kmeans programs on clusters of size 0 to 50, in increments of 1, and averaged each data point over five runs.