Distributed System Lab 4

Name: Guanyu Wang Email: guanyuw@andrew.cmu.edu

Name: Zeyuan Li Email: zeyuanl@cmu.edu

# Approaches to parallel the program

The parallel version of k-means using MPI is to initiate multiple processes based on a single piece of code. The pseudo code is like follows:

(1) Devide all data into equal pieces by rank=0 process and MPI\_Send to all other processes

(2) MPI\_Bcast initial cluster centroids and dimensions

(3) For each process, assign labels to each of the data points (find membership)

(4) MPI\_Allreduce the sum of each element in the cluster centroid and cluster size and for all processes to get the total sum of each element in vector of cluster centroid and total cluster size

(5) For each process, calculate the new cluster centroid.

(6) Repeat (3)-(5) until converge

(7) MPI\_Reduce the running time for k-means and for the subsystem as a whole to find the longest time as the system k-means and system total time respectively.

## Parallelize main driver

The main driver program running on each process calls kmeans\_read() to get the its subset of input data. Then rank=0 process broadcast the initial cluster centroids to all other processes. This is followed by the main kmeans() on each processes. Finally, we do some clean up and reduce to get the longest time as the whole system running time.

## Parallelize read input data

The kmeans\_read() step is different on rank=0 process and all other processes. The rank=0 process read the whole input file and divide the input as near equal sizes. Then rank=0 process uses MPI\_Send to send each chunk of data to respective process. The all other processes just wait their and use MPI\_Recv to get the input data for them to process.

## Parallelize k-means

## Parallelize write output data

# Experimentation and analysis

## Performance graph

## Performance analysis