Michael Burdi [maburdi@csu.fullerton.edu](mailto:maburdi@csu.fullerton.edu)

Bijaya Shrestha [sthavjay@csu.fullerton.edu](mailto:sthavjay@csu.fullerton.edu)

 Giuliana Pham [gpham@csu.fullerton.edu](mailto:gpham@csu.fullerton.edu)

CPSC 474 -02

Dr. D. Bein

Project 2 Submission

PSEUDOCODE FOR OUR PROGRAM

1. ***Pseudocode for main.cpp***
   1. Implemented 3-vector under the struct
   2. Calculated the square Euclidean distance between two points (vectors)
   3. Calculated the distance index of all objects in a process.

|  |
| --- |
| \* param objects |
| \* param N - number of objects |
| \* param centroids |
| \* param K - number of clusters (centroids) |
| \* return the distance index |

* 1. Load in data from a text file where every line contains 3 numerical integer values separated by spaces. The initial expected size of the data. If data is larger, then the array will be reallocated with {capacity} more elements. And return the exact number of items read from the file.
  2. Load the input file “.txt” under the function loadInputData.
  3. In main function, loading MPI components like init, finalize, Comm\_ranks and so on.
  4. User has to input a number of cluster while running a program like 3 or 4. But the number of clusters shouldn’t be more than the number of vectors per each process.
  5. Created an error function is error occurs and Terminate a process with the abort.
  6. Distributed the items among processes and give one more at a time to processes where number is not evenly divisible using “for loop”.
  7. Created functions to Allocate memory for each process, distribute vectors among the processes, and wait for everyone, Each process selects K out of N objects to be centroids.
  8. Created a loop to calculate the number of needed cluster and found the average of them.
  9. Made the first K as the center to start and find the other closest centroid.
  10. Again, found the actual centroid by finding out the mean of all.
  11. Run the loop to the “N” process forming number of “K” cluster.
  12. Create a barrier to meet up all the process at the end

Return

|  |
| --- |
|  |

*Output:*

Number of clusters: 5

Process 0 count = 13

Process 2 count = 14

Process 1 count = 13

P1,0 <-91,1,75>

P1,1 <-11,-17,-9>

P1,2 <-93,6,37>

P1,3 <-23,-9,16>

P1,4 <-72,72,35>

J' = 991189.000000

J = 840469.000000

Threshold = 0.100000

P2,0 <-77,1,-48>

P2,1 <4,-12,0>

P2,2 <-94,10,72>

P2,3 <46,-41,-72>

P2,4 <-61,-67,62>

J' = 1205645.000000

J = 1074841.000000

Threshold = 0.100000

P0,0 <-50,33,-49>

P0,1 <-14,6,-14>

P0,2 <64,-31,43>

P0,3 <-82,-26,-38>

P0,4 <92,1,44>

J' = 1117343.000000

J = 1036269.000000

Threshold = 0.100000

P0,0 <-50,33,-49>

P0,1 <-15,0,-11>

P0,2 <64,-31,43>

P0,3 <-82,-26,-38>

P0,4 <92,1,44>

J' = 1036269.000000

J = 1036219.000000

Threshold = 0.100000

P1,0 <-91,1,75>

P1,1 <-13,-24,-20>

P1,2 <-93,6,37>

P1,3 <-23,-9,16>

P1,4 <-72,72,35>

J' = 840469.000000

J = 848061.000000

Threshold = 0.100000

P2,0 <-77,1,-48>

P2,1 <-2,-9,-4>

P2,2 <-94,10,72>

P2,3 <46,-41,-72>

P2,4 <-61,-67,62>

J' = 1074841.000000

J = 1073665.000000

Threshold = 0.100000

P0,0 <-50,33,-49>

P0,1 <-15,-1,-10>

P0,2 <64,-31,43>

P0,3 <-82,-26,-38>

P0,4 <92,1,44>

J' = 1036219.000000

J = 1036299.000000

Threshold = 0.100000

P2,0 <-77,1,-48>

P2,1 <-3,-9,-5>

P2,2 <-94,10,72>

P2,3 <46,-41,-72>

P2,4 <-61,-67,62>

J' = 1073665.000000

J = 1073581.000000

Threshold = 0.100000

P2,0 <-77,1,-48>

P2,1 <-3,-9,-5>

P2,2 <-94,10,72>

P2,3 <46,-41,-72>

P2,4 <-61,-67,62>

J' = 1073581.000000

J = 1073581.000000

Threshold = 0.100000

Process 0 centroids

<-50,33,-49>

<-15,-1,-10>

<64,-31,43>

<-82,-26,-38>

<92,1,44>

Process 1 centroids

<-91,1,75>

<-13,-24,-20>

<-93,6,37>

<-23,-9,16>

<-72,72,35>

Process 2 centroids

<-77,1,-48>

<-3,-9,-5>

<-94,10,72>

<46,-41,-72>

<-61,-67,62>