LITERATURE REVIEW: Performance Comparison of Centroid Based Clustering Algorithms

Aagyapal Kaur
School of Computer Science
Carleton University
Ottawa, Canada K1S 5B6
aagyapalkaur@cmail.carleton.ca

October 28, 2021

1 Introduction

Enterprises today are dealing with the massive size of data, which have been explosively increasing. The key requirements to address this challenge are to extract, analyse, and process data in a timely manner. Clustering is an essential data mining tool that plays an important role for analysing the big data. However, large scale data clustering has become a challenging task because of the large amount of information that emerges from technological progress in many areas, including finance and business informatics.

Patterns identification groups such as feature vectors ,data items or other observation done by supervised or unsupervised classification method, Clustering is belongs to the method of unsupervised classification method .Objects with similar features are grouped to form a cluster. Clustering has wide intrigue and has been applied to address many field like data compression dimension reduction, feature selection, dimension reduction and vector quantization. Requirement of an application decide the opinion of quality cluster. There are a few strategies for discovering cluster utilizing techniques of splitting-merging, neural networks, and distribution – based clustering. Centroid-based clustering arranges the data into non-hierarchical clusters, in contrast to hierarchical clustering. In centroid –based clustering there are many types among that K-means is the most widely-used one. Centroid-based algorithms are efficient in grouping but sensitive to outliers and initial conditions[5].

K-means is a well-known clustering algorithm for its simplicity and easy implementation. K-means was ranked second of top 10 algorithms in data mining by the ICDM Conference in October 2006, while C4.5 was ranked first [10]. Compared with other clustering algorithms, K-means algorithm has three major advantages covering simple implementation, efficient when handling a large data sets and a solid theoretical foundation based on the greedy optimization of Voronoi partition[7].

In addition, MPI (Message Passing Interface) is a library specification for message passing, which is proposed as a standard program model. MPI is an application programmer interface of message passing, together with protocol and semantic specifications for how its features must behave in any implementation[4]. MPI aims to maintain the high performance, scalability and portability.

In this project, I mainly focused on performance analysis of Centroid Technique based clustering algorithms that include K-means, K-means++ and Parallel K-means using Messaging Passing Interface i.e. MPI and would like to find out which one of these is effectively performed on the datasets.

2 LITERATURE REVIEW

2.1 K-means Clustering Algorithm

K-means is one of the most famous partitional algorithms in cluster process. K-means has a rich and various history as it was autonomously found in various logical fields by Steinhaus (1956), Lloyd (proposed in 1957, distributed in 1982), Ball and Hall (1965), and MacQueen (1967) . K-means is a method of data analysis. The objective is to partition N data objects into K clusters (K<N) such that the objects in the same cluster are as similar as possible and a dissimilar as possible in different clusters[5].

Algorithm1: K-Means clustering algorithm
The main steps of K-means algorithm are as follows[6] :

1. Select k points as initial centroids arbitrarily.
2. Calculate the distance (Euclidean Distance) between the rest data and k centroids. Arrange each data into the nearest cluster. According to Euclidean Distance formulae, the distance between the two points in the plane with coordinates (x_1, y_1) and (x_2, y_2) is given by:

$$\mathrm{d} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

3. After processing all the points, compute the average of all the records in each same cluster.
4. The average of each cluster is used as the new centroid.
5. Calculate the difference between the new centroid with the original one in the same cluster. If the difference is smaller than the threshold or the number of iterations of the algorithm has been reached for the maximum, the algorithm is over. Otherwise, the new centroid is substituted for the original ones. Return (2) and continue until the convergence is achieved and obtain the final cluster .

The K-means algorithm requires three user-specified parameters: number of clusters K, cluster initialization, and distance metric. The most critical choice is K. Typically, K-means is run independently for different values of K and the partition that appears the most meaningful to the domain expert is selected. Different initializations can lead to different final clustering because K-means only converges to local minima. One way to overcome the local minima is to run the K-means algorithm, for a given K, with multiple different initial partitions and choose the partition with the smallest squared error. K-means is typically used with the Euclidean metric for computing the distance between points and cluster centers. As a result, K-means finds spherical or ball-shaped clusters in data.

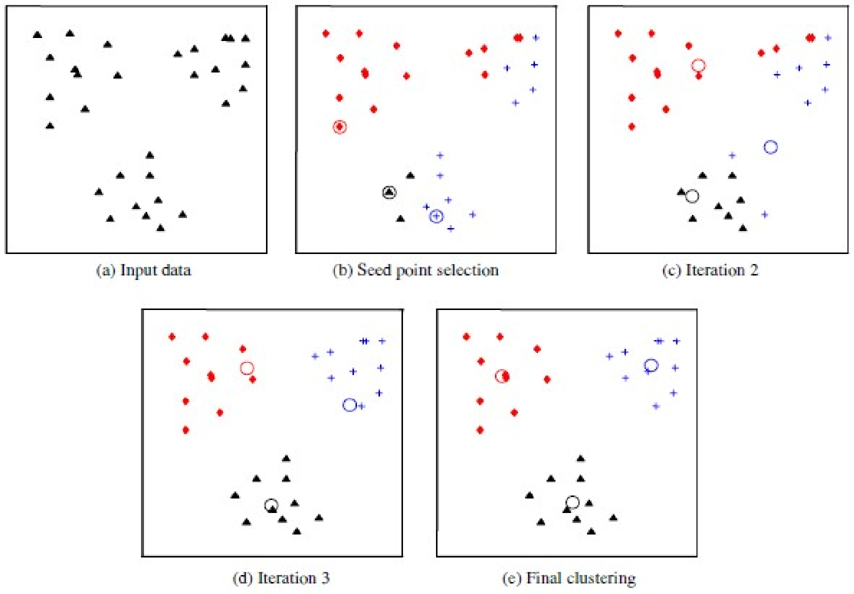


Figure 1: Illustration of K-means algorithm. (a) Two-dimensional input data with three clusters; (b) three seed points selected as cluster centers and initial assignment of the data points to clusters; (c) and (d) intermediate iterations updating cluster labels and their centers; (e) final clustering obtained by K-means algorithm at convergence.[5]

2.2 Limitation of K-means:

The main critical problem of this algorithm is that it might be blocked locally based on the initial random chosen centers. The k-means++ algorithm is developed to solve this problem, spreading out the initial centers with an updating non-uniform distribution[2].

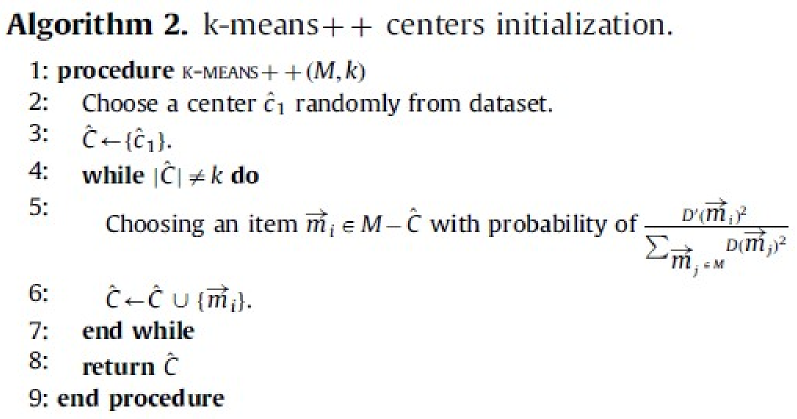
2.3 To overcome the limitation of Kmeans, we use K-means++ Algorithm:

The K-means++ algorithm is identical to k-means except the initialization of centers. K-means++ tries to choose a set of carefully selected initial centers instead of random initialization .This algorithm ensures a smarter initialization of centroids and ameliorate the quality of clustering. The k-means algorithm begins with an arbitrary set of cluster center. At any given time, let D'(m) denote the shortest distance from a data point m to the closest center we have already chosen, the initialization step of k-means++ can be defined as[2],[9]:

1. Choose an initial center c_1 uniformly at random from M where M is a set of datapoints.
2. Choose the next center c_i, selecting $c_i = m\epsilon M$ with probability proportional to

$$\frac{D'(m)^2}{\sum_{m\epsilon M} D(m)^2}$$

3. Repeat Step (2) until we have chosen a total of k centers
4. Proceed as with the standard k-means algorithm



Algorithm 2. k-means++ centers initialization.

```
1: procedure K-MEANS++$(M, k)$
2:     Choose a center $\hat{c}_1$ randomly from dataset.
3:     $\hat{C} \leftarrow \{\hat{c}_1\}$.
4:     while $|\hat{C}| \neq k$ do
5:         Choosing an item $\vec{m}_i \in M - \hat{C}$ with probability of $\frac{D'(\vec{m}_i)^2}{\sum_{\vec{m}_j \in M} D(\vec{m}_j)^2}$
6:         $\hat{C} \leftarrow \hat{C} \cup \{\vec{m}_i\}$.
7:     end while
8:     return $\hat{C}$
9: end procedure
```

Figure 2: K-mean++ Algorithm.[9]

2.4 Message Passing Interface

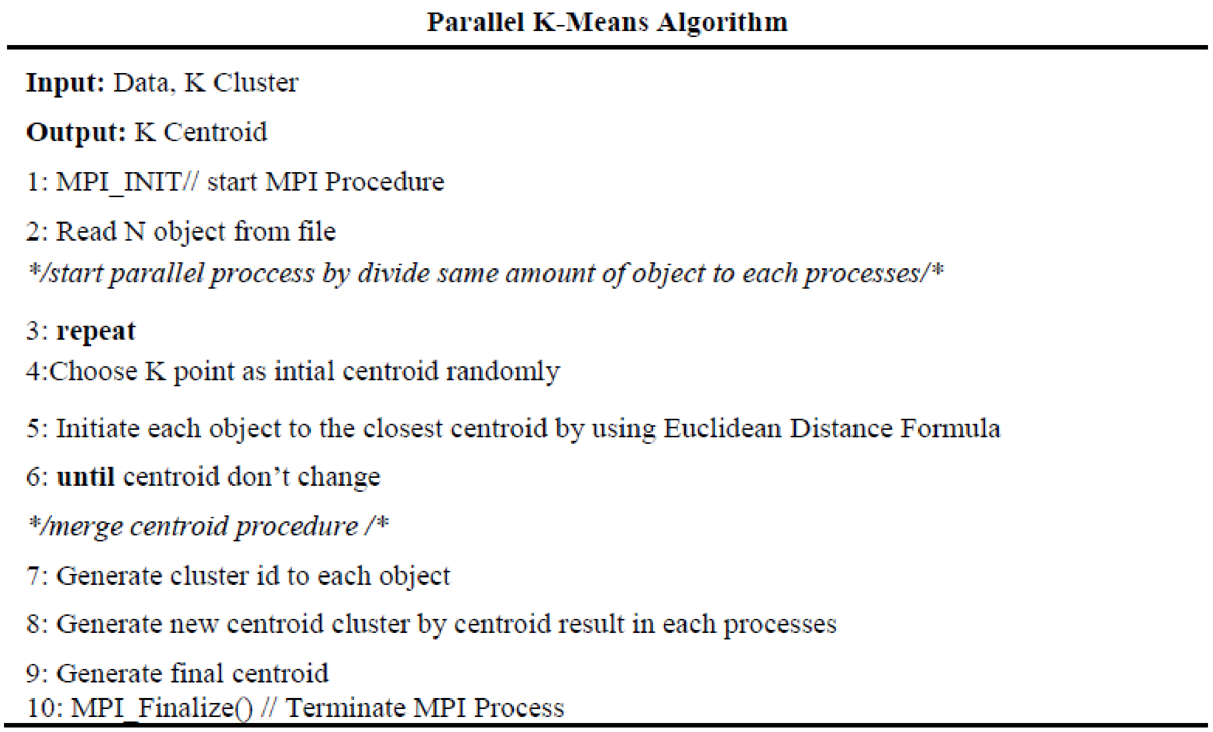
Message Passing Interface (MPI) is a standard developed by the Message Passing Interface Forum (MPIF). MPI is a standard library specification designed to support parallel computing in a distributed memory environment. The first version (version1.0) was published in 1994 and the second version (MPI-2) was published in 1997 [3]. Both point to point and collective communication are supported. MPI is an API library consisting of hundreds of function interfaces called by computers such as computer clusters communicate with one another in the parallel development environment. MPI is a language-independent communications protocol. FORTRAN, C and C++ can directly call the API library. The goals of MPI are high performance, scalability and portability.

The standards of MPI are as follows[1]:

1. Point-to-point communication
2. Collective operations
3. Process groups
4. Communication contexts
5. Process topologies
6. Bindings for Fortran 77 and C
7. Environmental management and inquiry
8. Profiling interface

2.5 Parallel K-means Algorithm

Parallelized in basic have same meaning in partition data so data can be execute in same time. In parallel system, the main idea is partition to each processes so the processes will have same amount of data and can do the processes in same time. Each computer can do the algorithm, and have centroid in each process. After that, the result of each process will be merged in head node[8].



Parallel K-Means Algorithm

```
Input: Data, K Cluster
Output: K Centroid
1: MPI_INIT// start MPI Procedure
2: Read N object from file
*/start parallel proccess by divide same amount of object to each processes/*
3: repeat
4:Choose K point as intial centroid randomly
5: Initiate each object to the closest centroid by using Euclidean Distance Formula
6: until centroid don't change
*/merge centroid procedure /*
7: Generate cluster id to each object
8: Generate new centroid cluster by centroid result in each processes
9: Generate final centroid
10: MPI_Finalize() // Terminate MPI Process
```

Figure 3: Parallel K-means Algorithm.[8]

The above Algorithm explain the processing flow of our parallel K means algorithm, which is the key algorithm in our work. This algorithm utilizes K-means and the MPI parallel framework, it is hence simple and portable. All initialization is implemented by the MPI_INIT function, which is the first call of MPI program. It is the first executable statement of all of the MPI program. To start the MPI environment, it means the beginning of the parallel codes. The MPI_FINALIZE function is the last call of MPI program, and it ends the running of MPI program. It is the last executable statement of MPI program, otherwise, results of the procedure is unpredictable. MPI_FINALIZE symbolizes the end of the parallel codes.In our Parallel Kmeans algorithm, the parallelism is implemented by the data parallelism. The parallel processing in Parallel Kmeans is consistent with that of K means. Data objects are evenly partitioned in all processes and cluster centroids are replicated. The global operation for all cluster centroids is performed at the end of each iteration in order to generate new cluster centroids. Finally, output the clustering results: K centroids, I/O time and clustering time[11],[8].

References

[1] Yukiya Aoyama, Jun Nakano, et al. *Rs/6000 sp: Practical MPI programming.* IBM Poughkeepsie, New York, 1999.

[2] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding, p 1027–1035. *Society for Industrial and Applied Mathematics*, 2007.

[3] William Gropp and Ewing Lusk. Implementing mpi: The 1994 mpi implementors' workshop. In *Proceedings Scalable Parallel Libraries Conference*, pages 55–59. IEEE, 1994.

[4] William Gropp, Ewing Lusk, Nathan Doss, and Anthony Skjellum. A high-performance, portable implementation of the mpi message passing interface standard. *Parallel computing*, 22(6):789–828, 1996.

[5] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[6] Anil K Jain and Richard C Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[7] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[8] Fhira Nhita. Comparative study between parallel k-means and parallel k-medoids with message passing interface (mpi). *International Journal on Information and Communication Technology (IJoICT)*, 2(2):27–27, 2016.

[9] Saeed Shahrivari and Saeed Jalili. Single-pass and linear-time k-means clustering based on mapreduce. *Information Systems*, 60:1–12, 2016.

[10] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, S Yu Philip, et al. Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008.

[11] Jing Zhang, Gongqing Wu, Xuegang Hu, Shiying Li, and Shuilong Hao. A parallel k-means clustering algorithm with mpi. In *2011 Fourth International Symposium on Parallel Architectures, Algorithms and Programming*, pages 60–64. IEEE, 2011.