Pillar K-Means Clustering Algorithm Using MapReduce Framework

Pillar K-Means Clustering Algorithm Using MapReduce Framework

A L Ramdani[1], H B Firmansyah[2]

[12] Department of Informatic, Institut Teknologi Sumatera, Indonesia
Email: {[1]ahmadluky, [2]hafiz.budi}@if.itera.ac.id

Abstract. Pillar K-means cluster is one of clustering algorithm developed from K-means algorithm which is more effective than another similar algorithm. The advanced development of electronic data has brought two main impacts in data clustering algorithm, including how to store big data and how to process this type of data. This paper discussed the analysis and implementation of Pillar K-means algorithm in a distributed system using MapReduce Framework. We cited the existing Pillar K-Means Algorithm and implemented it by using MapReduce Framework. Various functions had been implemented simultaneously such as Mapper and Reducer, which are part of MapReduce Framework. The result showed that there was a positive performance regarding efficiency and scalability of Pillar K-means by using synthetic datasets.

1.Introduction

Clustering is one of a popular unsupervised-learning algorithm which is used widely in numerous aspect of lives [1]. Clustering is a process where an object is classified into a group by searching similarities between the objects [2]. The primary purpose of this algorithm is to determine an object group without a label needed. A cluster has a high similarity degree with same cluster and low similarity degree with other clusters. A clustering algorithm is implemented for several purposes such as image analysis, market analysis, data mining and pattern recognition

K-means is one of clustering algorithm which is popular compared with others algorithm. [3], K-means algorithm is also categorized as unsupervised-learning algorithm [3]. Moreover, K-means algorithm is ranked as a top 10 algorithm in data mining [4].

In recent years, cluster analysis has developed significantly. There are several types of algorithm developed which has a purpose to optimize algorithm performance, for example, K-means++ and Pillar K-means. Those two algorithms are trying to enhance K-means algorithm performance by approaching an object initialization as cluster centroid. Pillar algorithm had successfully shown excellent performance and been capable of becoming a solution regarding data outliers. However, there is still a problem of data with high dimension and big size which influences time complexity.

The research proposed a parallel computing algorithm as a solution to time complexity[5,6,7]. All of those algorithms had, for instance, limited programming model and parallel computing automation. This assumption is quite expensive for a big dataset with millions of objects. Then, it is vitally important to develop a data-driven clustering algorithm

MapReduce framework is a programming model which can be potentially used for processing big data in cluster computers[8,9]. MapReduce is divided up into two parts, map and reduce, which are usually implemented at functional programming. Map function process every input in order to produce temporary key and value pairs. Meanwhile, reduce function unite all temporary values according to their temporary keys. This programming model enables data processing on a cluster computer. Apache Hadoop is an example software of MapReduce programming model[10]

The main contribution of this paper is:
(1) Adapting Pillar K-means algorithm on MapReduce framework which had implemented Hadoop to create clustering method on big data.
(2) Tackling the problem of inefficient time complexity in grouping big data

This paper is organized as follows: section 2 will discuss Pillar K-means algorithm based on MapReduce framework. Section 3 will cover the experiment results and evaluate the algorithm. Finally, Section 4 will conclude the research

2.Pillar K-Mean Algorithm Base On MapReduce
This section will describe the main design of Pillar K-means algorithm which is implemented by using MapReduce framework. Firstly, it will generally explain Pillar K-means algorithm and analyze which parts that can distribute the process on a cluster computer. Then, it will elaborate on how we designed Pillar K-means algorithm on MapReduce framework.

2.1. Pillar K-Means Algorithm
K-means is an algorithm which can divide a group object into a not overlap sub-group object. It means that every object is precisely mapped to a cluster[12]. K-means use centroid to determine how many clusters identified in data. This algorithm is considered as quick, simple and effective to address data clustering problem[3].

K-means algorithm frequently conducts computation to calculate the distance between centroid and data object. On each iteration, it needs n x k to measure the distance, where n is the number of objects and k is the number of clusters created. Consequently, it always computes the distance between one object with irrelevant centroid and relevant. In general, the distance measurement computation needs more time to distribute data on a cluster computer.

K-means also has another down side concerning centroid initialization. In the beginning, a centroid is randomly chosen so that the clusters become unstable. The second problem is the number of clusters must be thoroughly observed to get the optimal cluster. One of an alternative solution that we can do is by implementing Pillar algorithm

Pillar algorithm is inspired by the construction of pillars at a building. The pillars are placed on the corner of the building to distribute the force. The same idea is adapted to process data cluster. In data clustering, an initial centroid is guessed on the corner of data or outside part of data

The main purpose of pillar algorithm is to choose the farthest objects which can then be chosen as initial centroid in the beginning. The most frequent measurement in Pillar algorithm is the distance between a data object and middle value on initial centroid. Also, it computes the number of objects and the farthest object as the neighbor of the centroid. This mechanism is performed to get an equal number of objects in a cluster and to prevent outlier to be selected as the centroid.

2.2. Pillar K-Means Algorithm Based On MapReduce
Based on the above discussion, Pillar k-means algorithm design needs two processes, including MapReduce pillar algorithm and MapReduce k-means algorithm.

2.2.1. MapReduce Pillar Algorithm
There are two steps in determining MapReduce pillar algorithm: choosing initial centroid and the next centroid candidate. Those two processes have one MapReduce job which consists of one map function and reduce. In initial centroid step, map function calculates the distance between every object to data centroid. Meanwhile, reduce function sorts objects based on the distance with data centroid. Initial

centroid pseudocode can be written as follows (Algorithm 1).

Algorithm 1. MapReduce Pillar – Centroid Initialization
Map Function (key, value)
Input: Global variable (mean), key and the sample value
Output: <key, value> pair, *where key is an index object, value is the minimum distance between object and mean*

```
1.  key: index objek
2.  value: object pada data
3.  Begin
4.    index = 1
5.    For i=0 to value.length do
6.      D1 = ComputeDistance (value[i], mean)
7.      key = index;
8.      value = D1;
9.    End For
10.   output <key, value> pair;
11. End
```

Fungsi Reduce (key, V)
Input: key is index of object, V is list of value distance (D1)
Output: <key, value> pair, key is an index object, value is the minimum distance of sorted data object

```
12. key: index
13. V: list of value
14. Begin
15.   while (list of value.Next()){
16.     D2= SortDistance(list of value.Next())
17.   }
18.   value = D2;
19.   output<key, value> pair;
20. End
```

Data used in map function is extracted from HDFS [14] file which is a pair (key, value) and each of them represents a data row respectively. Key as a value that represents object identity on data. Map Function Data can be broken up into several parts and distributed to each computer. Then, the computational process can be conducted in parallel on each computer. The output from map function is a pair of object identification data and data object distance with a mean value (D1). Reduce Function. The output of map function is used for reduce function, which is a pair of object identity and distance. In reduce function, there was a process of sorting the distance, and then it resulted in a loss of value descending-sorted object identity and distance (D2).

Algoritma 2. MapReduce Pillar – Selected centroid candida
Fungsi Map (key, value)
Input: Global variable *nmin, nbdis*, *k*, D2, key and the sample value
Output: <key, value> pair, key is index object, *value* is object of data to be candidate centroid (D2).

```
1.  key: index objek
2.  value: object of data
3.  Begin
4.    index = 1
5.    For i=0 to D2.length do
6.      For j=0 to value.length do
7.            DM = ComputeDistance (D2[i], value[j])
```

```
8.        End For
9.        If DM.max < nbdis and DM.count >= nmin
10.               D = D2[i]
11.               index = i
12.       End If
13.       If D.count == k
14.               break;
15.       End If
16.    End For
17.    key = index;
18.    value = D
19.    output <key, value> pair;
20. End
Fungsi Reduce (key, V)
Input: key is index of object, V is list of value object data (D)
Output: <key, value> pair, key is index object, value is distance value of object data that has been
sorted
21. key: index
22. V: list of value object data
23. Begin
24.    output<key, value> pair;
25. End
```

The further step in MapReduce pillar algorithm is choosing centroid candidate. In this step, the algorithm used nmin, nbdis, k, and D2 value. Nmin is a minimal number of neighbor objects from the centroid, nbdis is a maximal neighbor objects distance from centroid and k is the number of clusters. The value of nmin and nbdis determined the number of proportional objects in a cluster and to prevent choosing outlier as the centroid. Both of that value highly depended on the value of α and β [13]. The pseudocode in this step can be seen in Algorithm 2. Data that was used in map function was the result of a precedent step (D2). It was the group of the farthest objects. In the map function, there was a computational process of distance measurement between the object at D2 with every data object. Based on nmin, mbdis and k value, the process gathered object on D2 which was centroid candidate (k)

2.2.2. Algorithm MapReduce K-Means

On this paper, K-means MapReduce algorithm demonstrated a useful result in processing big data[11]. This type of algorithm used three main functions such as map, combine and reduce. Map function played a role as a function which conducted computing procedure for each data object which was near to centroid. Meanwhile, reduce function executed a procedure which updated new centroids. In order to minimize network communication, combine function grouped temporary <key, value> to a similar map function.

3.Experimental

3.1 Experimental environment

The analysis of algorithm performance which was conducted on Raspberry Pi cluster as a substitution of Personal computer (PC), consisted of four Raspberry Pi which had Broadcom BCM2837BO 1.4 GHz processor, 1 GB RAM, ethernet speed up to 300 Mbps and 8 GB storage. Hadoop (v2.7.5) and Java(v.1.8) were also used to implement master and slave nodes architecture. Raspberry Pi cluster can be seen at figure 1. Master node was responsible for saving knowledge and data processing. Meanwhile, the slave node was used to save physical data with a specific size block and conducting data processing as directed by a master node. Master node also made sure that the data processing activity was well-run without any error.

In this research, we used two different scenarios to evaluate algorithm performance. The first scenario was intended to analyze speed performance. By using this scenario, we used a multi-node approach with different data size and memory allocation run by Hadoop. The second scenario had an objective to measure the influence of the map function number regarding speed performance. This scenario was run on one node with same data variation on the first scenario. The change on map function number was extracted from a file called mapred-site.xml

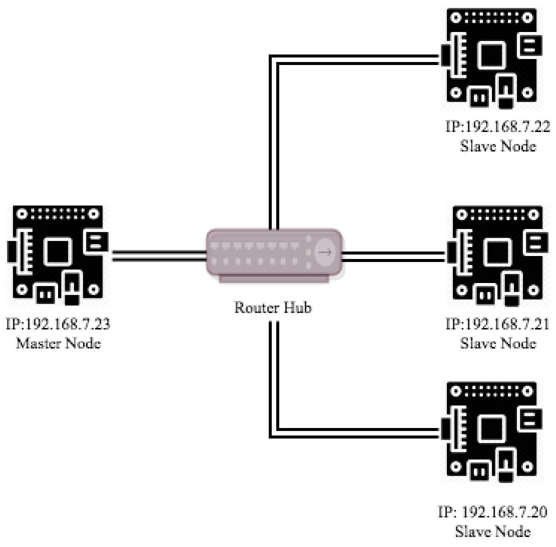

Figure 1. Cluster architecture using Rasberry Pi

3.2 Data

We used synthetic data which had Gaussian distribution. A synthetic process used sklearn library running on python sklearn [15]. The size of data was respectively showed as follows: 50 MB, 200 MB, 500 MB, 800 MB, and 900 MB

3.3 Result and discussion

The result of performance analysis based on the first scenario was shown in figure 2. The experiment of the first scenario was conducted ten times. The time value (figure 2) is the average of the experiment. Figure 2 also shows the change of node number on each measurement experiment which could be varied and able to influence algorithm computational speed pillar k-means. This condition happened because every node used their respective resource in processing data

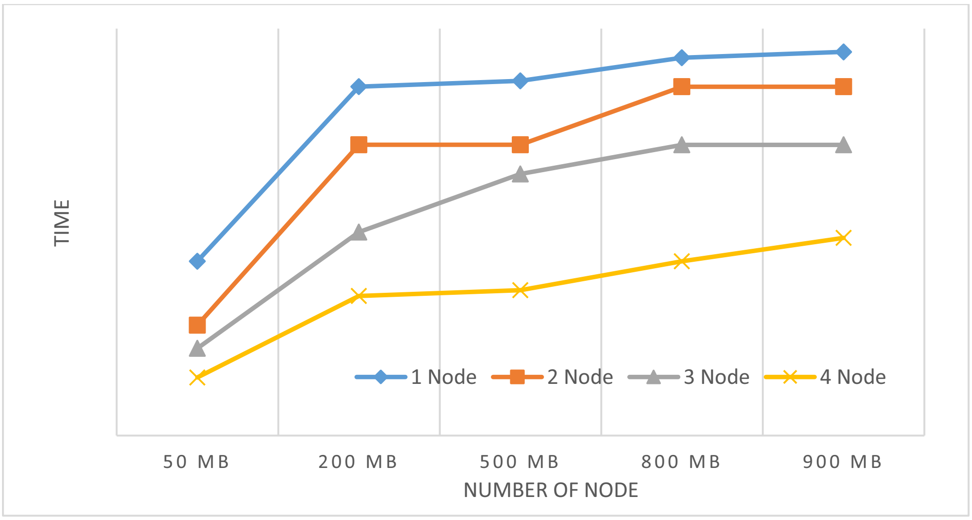

Figure 2. Effect number node to computational speed

The second scenario was organized by changing the number of map function. This experiment was conducted ten times by using map function 2, 4 and 6. The time value which was on figure 3 was the average of an experiment. The result of this experiment was shown in figure 3. On Figure 3 explained that the algorithm computational speed performance was faster if we used four map function compared with 2 and six because Raspberry Pi B had four core processor then by using four map function enable us to optimize resource utilization and preventing process queue.

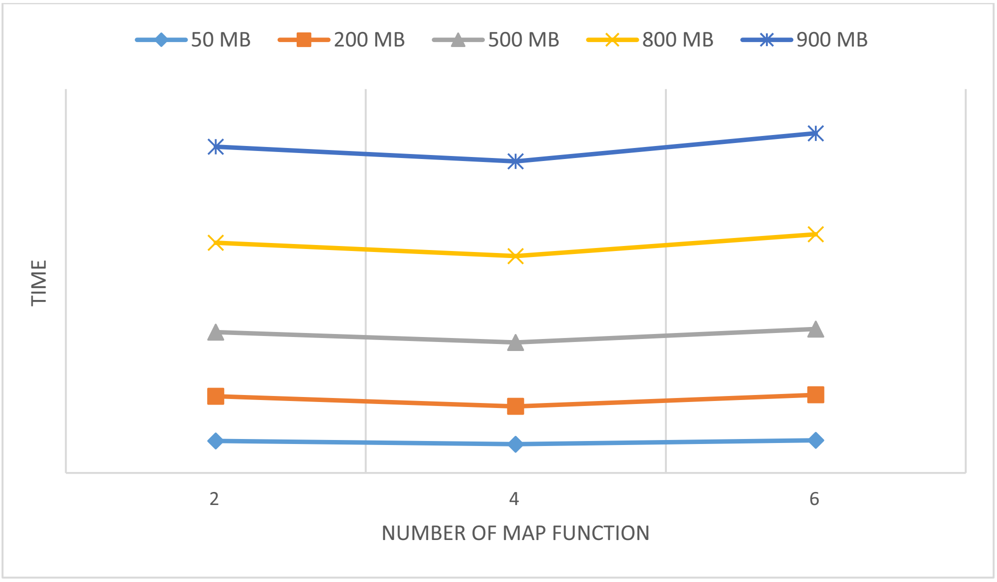

Figure 3. Effect number function map to computational speed

4.Conclusion

Clustering algorithm is one of an unsupervised-learning algorithm which is used broadly in various fields. In this research, we proposed the development of pillar k-means algorithm which runs on MapReduce framework. The process of algorithm development used Hadoop. The results conducted from two scenarios showed that the use of MapReduce framework on pillar k-means algorithm was able to significantly enhance computational speed by adding the number of nodes. Besides, to determine the number of suitable map function with Hadoop configuration can enhance computational speed if number core processor same as number map function. However, this research needs to be more explored especially by manipulating resources such as CPU and Disk I/O to observe energy consumption

Acknowledgments
This research is supported by Hibah Mandiri (Grant) of Institut Teknologi Sumatera in 2018.

References

1) J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, 2000
2) V. Estivill-Castro, “Why So Many Clustering Algorithms-A Position Paper”, SIGKDD Explorations, 2002, Vol. 4, No. 1, pp. 65-75.
3) J. B. MacQueen, “Some Methods for Classification and Analysis of Multivariate Observations”, Berkeley: University of California Press, 1967, pp. 281-297.
4) X. Wu, V. Kumar, J.-R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. Mclachlan, A. Ng, B. Liu, P. Yu, Z.-H. Zhou, M. Steinbach, D. Hand, and D.Steinberg, “Top 10 Algorithms in Data Mining”, Knowledge and Information Systems, 2008, Vol. 14, No. 1, pp. 1-37

5) Rasmussen, E.M., Willett, P.: Efficiency of Hierarchical Agglomerative Clustering Using the ICL Distributed Array Processor. Journal of Documentation 45(1), 1–24 (1989)
6) 2. Li, X., Fang, Z.: Parallel Clustering Algorithms. Parallel Computing 11, 275–290 (1989)
7) 3. Olson, C.F.: Parallel Algorithms for Hierarchical Clustering. Parallel Comput- ing 21(8), 1313–1325 (1995)
8) Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clus- ters. In: Proc. of Operating Systems Design and Implementation, San Francisco, CA, pp. 137–150 (2004)
9) 5. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clus- ters. Communications of The ACM 51(1), 107–113 (2008)
10) http://hadoop.apache.org/
11) Zhao, Weizhong, Huifang Ma, and Qing He. "Parallel k-means clustering based on MapReduce." *IEEE International Conference on Cloud Computing*. Springer, Berlin, Heidelberg, 2009.
12) Witten, Ian H., et al. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2016.
13) Barakbah, Ali Ridho, and Yasushi Kiyoki. "A pillar algorithm for k-means optimization by distance maximization for initial centroid designation." Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on. IEEE, 2009.
14) Shvachko, Konstantin, et al. "The Hadoop distributed file system." Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on. Ieee, 2010.
15) http://scikit-learn.org/