MapReduce Design of K-Means Clustering Algorithm

Prajesh P Anchalia
Department of CSE,
R V College of Engineering,
Bangalore, India
Email: prajeshanchalia@yahoo.in

Anjan K Koundinya
Department of CSE,
R V College of Engineering,
Bangalore, India
Email: annjank2@gmail.com

Srinath N K
Department of CSE
R V College of Engineering,
Bangalore, India
Email: srinath_nk@yahoo.com

***Abstract*— Cluster is a collection of data members having similar characteristics. The process of establishing a relation or deriving information from raw data by performing some operations on the data set like clustering is known as data mining. Data collected in practical scenarios is more often than not completely random and unstructured. Hence, there is always a need for analysis of unstructured data sets to derive meaningful information. This is where unsupervised algorithms come in to picture to process unstructured or even semi structured data sets by resultant. K-Means Clustering is one such technique used to provide a structure to unstructured data so that valuable information can be extracted. This paper discusses the implementation of the K-Means Clustering Algorithm over a distributed environment using Apache™ Hadoop. The key to the implementation of the K-Means Algorithm is the design of the Mapper and Reducer routines which has been discussed in the later part of the paper. The steps involved in the execution of the K-Means Algorithm has also been described in this paper based on a small scale implementation of the K-Means Clustering Algorithm on an experimental setup to serve as a guide for practical implementations.**

***Index Terms*— K-Means Clustering, MapReduce , Hadoop, Data Mining, Distributed Computing.**

I. INTRODUCTION

Any inference that delineates an argument is an outcome of careful analysis of a huge amount of data related to the subject. So to facilitate a comprehensive and definitive correlation of data we apply methods of data mining to group data and derive meaningful conclusions. Data mining thus can be defined as subject that discovers data relations by applying principles of artificial intelligence, statistics , database systems and likewise. In addition to just analysis this facilitates data management aspects, data modeling, visualization, complexity considerations.

Distributed Computing is a technique aimed at solving computational problems mainly by sharing the computation over a network of interconnected systems. Each individual system connected on the network is called a node and the collection of many nodes that form a network is called a cluster.

Apache™ Hadoop[1] is one such open source framework that supports distributed computing. It came into existence from Google's MapReduce and Google File Systems projects. It is a platform that can be used for intense data applications which are processed in a distributed environment. It follows a Map and Reduce programming paradigm where the fragmentation of data is the elementary step and this fragmented data is fed into the distributed network for processing. The processed data is then integrated as a whole. Hadoop[1][2][3] also provides a defined file system for the organization of processed data like the Hadoop Distributed File System. The Hadoop framework takes into account the node failures and is automatically handled by it. This makes hadoop really flexible and a versatile platform for data intensive applications. The answer to growing volumes of data that demand fast and effective retrieval of information lies in engendering the principles of data mining over a distributed environment such as Hadoop. This not only reduces the time required for completion of the operation but also reduces the individual system requirements for computation of large volumes of data.

Starting from the Google File Systems[4] and MapReduce concept, Hadoop has taken the world of distributed computing to a new level with various versions of Hadoop that are now in existence and also under Research and Development. Few of which include Hive[5] , Zookeeper [6] ,Pig[7]. The data-intensity today in any field is growing at a brisk space giving rise to implementation of complex principles of Data Mining to derive meaningful information from the data.

The MapReduce structure gives great flexibility and speed to execute a process over a distributed Framework. Unstructured data analysis is one of the most challenging aspects of data mining that involve implementation of complex algorithms.

The Hadoop Framework is designed to compute thousands of petabytes of data. This is primarily done by downscaling and

978-1-4799-0604-8/13/$31.00 ©2013 IEEE

consequent integration of data and reducing the configuration demands of systems participating in processing such huge volumes of data. The workload is shared by all the computers connected on the network and hence increase the efficiency and overall performance of the network and at the same time facilitating the brisk processing of voluminous data.

This paper discusses the K-Means Algorithm design over a span of sections starting with the Introduction , section II deals with Clustering Analysis , section III talks about K- Means Clustering, section IV discusses the MapReduce paradigm, K-Means clustering using MapReduce paradigm and the algorithm to design the MapReduce routines is discussed in section V, section VI talks about the experimental setup, section VII talks about system deployment, section VII about Implementation of the K-Means Clustering on a distributed environment and section IX Concludes the paper.

II. Cluster Analysis

Clustering basically deals with grouping of objects such that each group consists of similar or related objects. The main idea behind clustering is to maximize the intra-cluster similarities and minimize the inter cluster similarities.

The data set may have objects with more than attributes. The classification is done by selecting the appropriate attribute and relate to a carefully selected reference and this is solely dependent on the field that concerns the user. Classification therefore plays a more definitive role in establishing a relation among the various items in semi or unstructured data set.

Cluster analysis is a broad subject and hence there are abundant clustering algorithms available to group data sets. Very common methods of clustering involve computing distance, density and interval or a particular statistical distribution. Depending on the requirements and data sets we apply the appropriate clustering algorithm to extract data from them.

Clustering has a broad spectrum and the methods of clustering on the basis of their implementation can be grouped into

- Connectivity Technique
  Example: Hierarchical Clustering
- Centroid Technique
  Example: K-Means Clustering
- Distribution Technique
  Example: Expectation Maximization
- Density Technique
  Example: DBSCAN
- Subspace Technique
  Example: Co-Clustering

Advantages of Data Clustering

- Provides a quick and meaningful overview of data.
- Improves efficiency of data mining by combining data with similar characteristics so that a generalization can be derived for each cluster and hence processing is done batch wise rather than individually.
- Gives a good understanding of the unusual similarities that may occur once the clustering is complete.
- Provides a really good base for nearest neighboring and ordination of deeper relations.

III. K-Means Clustering

K-Means Clustering is a method used to classify semi structured or unstructured data sets. This is one of the most commonly and effective methods to classify data because of its simplicity and ability to handle voluminous data sets.

It accepts the number of clusters and the initial set of centroids as parameters. The distance of each item in the data set is calculated with each of the centroids of the respective cluster. The item is then assigned to the cluster with which the distance of the item is the least. The centroid of the cluster to which the item was assigned is recalculated.

One of the most important and commonly used methods for grouping the items of a data set using K-Means Clustering is calculating the distance of the point from the chosen mean. This distance is usually the Euclidean Distance though there are other such distance calculating techniques in existence. This is the most common metric for comparison of points.

Suppose there the two points are defined as P = (x1(P), x2(P), x3(P) …..) and Q = (x1(Q), x2(Q), x3(Q) …..). The distance is calculated by the formula given by

$$d(P,Q) = \sqrt{((x1(P) - x1(Q))^2 + (x2(P) - x2(Q))^2 + ...)}$$
$$= \sqrt{\left(\sum_{j=1}^{p} (xj(P) - xj(Q))^2\right)}$$

The next important parameter is the cluster centroid. The point whose coordinates corresponds to the mean of the coordinates of all the points in the cluster.

The data set may or better said will have certain items that may not be related to any cluster and hence cannot be classified under them, such points are referred to as outliers and more often than not correspond to the extremes of the data set depending on whether their values or extremely high or low.

-The main objective of the algorithm is to obtain a minimal squared difference between the centroid of the cluster and the item in the dataset.

$$|x_i^{(j)} - c_j|^2$$

Where x_i is the value of the item and c_j is the value of the centroid of the cluster.

The Algorithm is discussed below:

- The required number of cluster must be chosen. We will refer to the number of clusters to be 'K'.
- The next step is to choose distant and distinct centroids for each of the chosen set of K clusters.
- The third step is to consider each element of the given set and compare its distance to all the centroids of the K clusters. Based on the calculated distance the element is added to the cluster whose centroid is nearest to the element.
- The cluster centroids are re calculated after each assignment or a set of assignments.
- This is an iterative method and continuously updated.

IV. MapReduce Paradigm

MapReduce is a programming paradigm used for computation of large datasets. A standard MapReduce process computes terabytes or even petabytes of data on interconnected systems forming a cluster of nodes. MapReduce implementation splits the huge data into chunks that are independently fed to the nodes so the number and size of each chunk of data is dependent on the number of nodes connected to the network. The programmer designs a Map function that uses a (key,value) pair for computation. The Map function results in the creation of another set of data in form of (key,value) pair which is known as the intermediate data set. The programmer also designs a Reduce function that combines value elements of the (key,value) paired intermediate data set having the same intermediate key. [10]

Map and Reduce steps are separate and distinct and complete freedom is given to the programmer to design them. Each of the Map and Reduce steps are performed in parallel on pairs of (key,value) data members. Thereby the program is segmented into two distinct and well defined stages namely Map and Reduce. The Map stage involves execution of a function on a given data set in the form of (key,value) and generates the intermediate data set. The generated intermediate data set is then organized for the implementation of the Reduce operation. Data transfer takes place between the Map and Reduce functions. The Reduce function compiles all the data sets bearing the particular key and this process is repeated for all the various key values. The final out put produced by the Reduce call is also a dataset of (key,value) pairs. An important thing to note is that the execution of the Reduce function is possible only after the Mapping process is complete.

Each MapReduce Framework has a solo Job Tracker and multiple task trackers. Each node connected to the network has the right to behave as a slave Task Tracker. The issues like division of data to various nodes , task scheduling, node failures, task failure management, communication of nodes, monitoring the task progress is all taken care by the master node. The data used as input and output data is stored in the file-system.

V. K-Means Clustering using MapReduce

The first step in designing the MapReduce routines for K-means is to define and handle the input and output of the implementation. The input is given as a <key,value> pair , where 'key' is the cluster center and 'value' is the serializable implementation of vector in the data set.

The prerequisite to implement the Map and Reduce routines is to have two file one that houses the clusters with their centroids and the other that houses the vectors to be clustered.

Once the set of initial set of clusters and chosen centroids is defined and the data vectors that are to be clustered properly organized in two files then the clustering of data using K-Means clustering technique can be accomplished by following the algorithm to design the Map and Reduce routines for K-Means Clustering.

The initial set of centers is stored in the input directory of HDFS prior to Map routine call and they form the 'key' field in the <key,value> pair. The instructions required to compute the distance between the given data set and cluster center fed as a <key,value> pair is coded in the Mapper routine. The Mapper is structured in such a way that it computes the distance between the vector value and each of the cluster centers mentioned in the cluster set and simultaneously keeping track of the cluster to which the given vector is closest. Once the computation of distances is complete the vector should be assigned to the nearest cluster.

Once Mapper is invoked the given vector is assigned to the cluster that it is closest related to. After the assignment is done the centroid of that particular cluster is recalculated. The recalculation is done by the Reduce routine and also it restructures the cluster to prevent creations of clusters with extreme sizes i.e. cluster having too less data vectors or a cluster having too many data vectors. Finally, once the centroid of the given cluster is updated, the new set of vectors and clusters is re-written to the disk and is ready for the next iteration.

After understanding of what the input, output and functionality of the Map and Reduce routines we design the Map and Reduce classes by following the algorithm discussed below.

Algorithm 1 Mapper design for K-Means Clustering

```
0: procedure KMEANMAPDESIGN
0:    LOAD Cluster file
0:    fp = Mapclusterfile
0:    Create two list
0:    listnew = listold
0:    CALL read (Mapclusterfile)
0:    newfp = MapCluster()
0:    dv =0
0:    Assign correct centeroid
0:    read(dv)
0:    calculate centerorid
0:    dv = minCenter()
0:    CALL KmeansReduce()
0: end procedure=0
```

Algorithm 2 Reducer design for K-Means Clustering

```
0: procedure KMEANREDUCEDESIGN
0:    NEW ListofClusters
0:    COMBINE resultant clusters from MAP CLASS.
0:    if cluster size too high or too low then
      RESIZE the cluster
0:       C_Max = findMaxSize(ListofClusters)
0:       C_min = findMinSize(ListofClusters)
0:       if C_max > (1/20) totalSize then Resize(cluster)
0:          WRITE cluster FILE to output DIRECTORY.
0:
```

Algorithm 3 Implementing KMeans Function

```
0: procedure KMEANS FUNCTION
0:    if Initial Iteration then LOAD cluster file from DIREC-
      TORY
0:    elseREAD cluster file from previous iteration
0:       Create new JOB
0:       SET MAPPER to map class defined
0:       SET REDUCER to reduce class define
0:       paths for output directory
0:       SUBMIT JOB
0:
```

VI. EXPERIMENTAL SETUP

The experimental setup consists of a three nodes sharing a private LAN via a managed switch. One of the Nodes is used as a Master which supervises the data and flow of control over all the other nodes in the Hadoop Cluster. The nodes used in the implementation are of similar configuration. All the nodes run on a Intel – Core 2 Duo processor.

The cluster used for the implantation has 7 nodes connected over a private LAN network. All the nodes use Ubuntu operating system with Java JDK 7, SSH installed on them. The Apache™ Hadoop bundle available on the official website was used to install Hadoop. The Single-Node and consequent Multi-Node Setup was accomplished by following the installation guide found at [8][9][10].

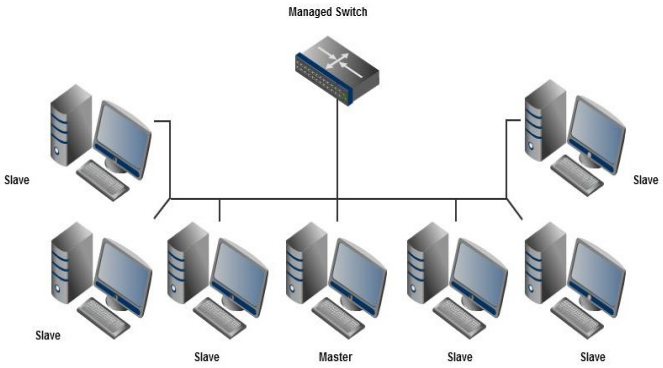

Figure 1: Experimental Setup of 7 Nodes.

VII. SYSTEM DEPLOYMENT

The visual deployment of the system is depicted in Figure 2. The Data set is fed to the master node. The data set is along with the Job is distributed among the nodes in the network. The Map function is called for the input in form of <key,value> pair. Once the assignment is complete, the Reduce function recalculates the centroids and makes the data set ready for the subsequent iterations

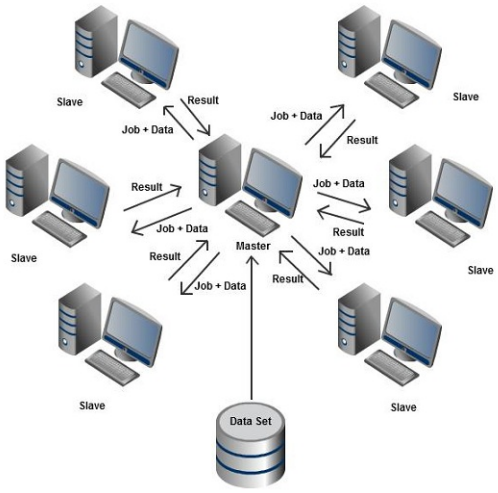

Figure 2: System Deployment

VIII. Implementation of The K-Means Algorithm on Distributed Network.

The K-Means Clustering Algorithm is implemented on a distributed network involves the following steps:

- The input data vectors and the initial set of chosen cluster centers is stored in the input directory of the HDFS and creating an output directory to house the result of clustered data.
- SSH into the master node, with the input directory containing the vectors and clusters run the program that is structured on the Algorithm used to define the Map and Reduce routines discussed in section V of this paper.
- Based on the iterations and the value of 'K' the resultant clusters of data can be found in the output directory which can be echoed to the output directory in HDFS.

IX. Conclusion and On Going Research Work

The volume of information exchange in today's world engages huge quantity of data processing. We through this paper have discussed the implementation of K-Means Clustering Algorithm over a distributed network. Not only does this algorithm provide a robust and efficient system for grouping of data with similar characteristics but also reduces the implementation costs of processing such huge volumes of data.

Data Mining being one of the most important tools in information retrieval. The rate of information exchange today is growing spectacularly fast and so there arises a need of processing huge volume of data. To respond to this need we try and implement the vital algorithms used for data mining on a distributed or a parallel environment to reduce the operational resources and increase the speed of operation.

Not matter how good anything can be there is always scope for improvement. Primary area of improvement in any algorithm is its accuracy. Research work has gone in since the first implementation of K-Means Algorithm in 1970's. Another area of improvement particular to the K-Means Algorithm is the selection of initial set of centroids. These two areas are of primary concern beside a few secondary issues. One of which is reducing the number of iterations required to complete a task this is mainly achieved by choosing the right set of initial centroids[12]. Another issue is to improve the algorithm scalability to support processing of high volume datasets, parallel implementation of K-Means Algorithm is an area of research that is aimed at addressing this issue. One more important issue is handling the outliers, though not related these points that are difficult to cluster and usually that make up the extreme cases cannot be ignored and hence different ways of clustering these outliers is another research hotspot. The number of iterations required to cluster data can be reduced by choosing the right set of initial set of centroids.

Acknowledgement

We would like to thank all the members of the Hadoop team present and past at RV College of Engineering for their hard work and valuable contribution towards the experimental setup. We also would like to thank the technical staff in the research lab at the Department of Computer Science and Engineering, RV College of Engineering for their ever available helping hand.

References

[1] Apache Hadoop. http://hadoop.apache.org/

[2] J. Venner, Pro Hadoop. Apress, June 22, 2009.

[3] T. White, Hadoop: The Definitive Guide. O'Reilly Media, Yahoo! Press, June 5, 2009.

[4] S. Ghemawat, H. Gobioff, S. Leung. "The Google file system," In Proc.of ACM Symposium on Operating Systems Principles, Lake George, NY, Oct 2003, pp 29–43.

[5] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu,P. Wyckoff, R. Murthy, "Hive – A Warehousing Solution Over a Map-Reduce Framework," In Proc. of Very Large Data Bases, vol. 2 no. 2,August 2009, pp. 1626-1629.

[6] F. P. Junqueira, B. C. Reed. "The life and times of a zookeeper," In Proc. of the 28th ACM Symposium on Principles of Distributed Computing, Calgary, AB, Canada, August 10–12, 2009.

[7] A. Gates, O. Natkovich, S. Chopra, P. Kamath, S. Narayanam, C.Olston, B. Reed, S. Srinivasan, U. Srivastava. "Building a High-Level Dataflow System on top of MapReduce: The Pig Experience," In Proc. of Very Large Data Bases, vol 2 no. 2, 2009, pp. 1414–1425

[8] Description if Single Node Cluster Setup at: http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/ visited on 21st January, 2012

[9] Description of Multi Node Cluster Setup at: http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-multi-node-cluster/ visited on 21st January, 2012

[10] Anjan K Koundinya, Srinath N K, A K Sharma, Kiran Kumar, Madhu M N and Kiran U Shanbagh,Map/Reduce Design anf Implementation of Apriori Algorithm for handling Voluminous Data-Sets, Advanced Computing: An International Journal (ACIJ), Vol.3, No.6, November 2012

[11] J. Dean and S. Ghemawat. Mapreduce: Simplied data processing on large clusters.Communications of the ACM, 51(1):107-113, 2008.

[12] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.H. Bae, J. Qiu, and G. Fox. Twister: A runtime for iterative mapreduce. In Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, pages 810-818. ACM, 2010.