A SYNOPSIS ON



Comparative Analysis of K-Means Clustering for Big Data Analysis Using MapReduce programing model



Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY IN

COMPUTER SCIENCE & ENGINEERING

Submitted by:

Anurag Semwal University Roll no- 2018206

Nipun Rawat University Roll no- 2018523

Rishav Bisht University Roll no- 2018636

**Under the Guidance of**

Dr. Satvik Vats



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GRAPHIC ERA HILL UNIVERSITY, DEHRADUN**

**2023-2024**

**Table of Contents**

Chapter No. Description Page No.

Chapter 1 Abstract 1

Chapter 2 Introduction and Problem Statement 2-3

Chapter 3 Background/ Literature Survey 2-3

Chapter 4 Objectives 1-2

Chapter 5 Hardware and Software Requirements 1

Chapter 6 Possible Approach/ Algorithms 2-4

References 1

Abstract

Huge amounts of structured and unstructured are being collected from various sources. These data called as Big Data which are difficult to handle by a single machine require the work to be distributed across many computers. Hadoop is a distributed framework which uses MapReduce programming model to process the data in a distributed manner. Clustering analysis is one of the important research areas in the field of data mining. Clustering is the most commonly used data processing algorithms. Clustering is a division of data into different groups. Data are grouped in such a way that data of the same group are similar and the data in other groups are dissimilar. Clustering aims in minimizing intra-class similarity and in maximizing interclass dissimilarity. k-Means is the popular clustering algorithm because of its simplicity .Nowadays, as the volume of data increases, researchers started to use MapReduce which is a parallel processing framework to get high performance. But, MapReduce is unsuitable for iterated algorithms owing to repeated times of restarting jobs, big data reading and shuffling. To overcome this problem, a novel processing model in MapReduce called optimized k means clustering method which uses the methods of probability sampling and clustering, merging using two algorithms called weight based merge clustering and distribution based merge clustering is introduced to eliminate the iteration dependence and obtain high performance. Also the algorithms are compared with the k-medoids clustering algorithms

Problem Statement : Comparative Analysis of K-Means Clustering for Big Data Analysis Using MapReduce programing model

1. Introduction

The common thing among all computing devices is the potential to generate data. The popularity of the Internet along with a sharp increase in the network bandwidth available to users has resulted in the generation of huge amounts of data. The data may also be too big to store on a single machine. To reduce the time it takes to process the data, and to store the data, it is necessary to write programs that can execute on two or more computers and distribute the workload among them. To address the above issues, Google developed the Google File System (GFS), a distributed file system architecture model for large-scale data processing and created the MapReduce programming model. The MapReduce programming model is a programming abstraction that hides the underlying complexity of distributed data processing. Therefore the difficulty of parallelizing computation, distribute data and handle faults no longer become an issue. This is because the MapReduce framework handles all these details internally and removes the responsibility of having to deal with these complexities from the programmer. A novel processing model in MapReduce is proposed in this paper, in which sampling is used to eliminate the iteration dependence of k-means and to obtain high performance. In particular, the contributions of the paper are summarized as follows. First, a novel method to optimize k-means clustering algorithms using MapReduce is presented, which eliminates the dependence of iteration and reduces the computation cost of algorithms. The implementation defines the mapper and reducer jobs and requires no modifications to the MapReduce framework. Second, two sample merging strategies for the processing model is proposed and extensive experiments are conducted to study the effect of various parameters using a real dataset. The results show that the proposed methods are efficient and scalable.

Data analysis underlies many computing applications, either in a design phase or as part of their operations. Data analysis procedures can be bifurcated as either exploratory or confirmatory based on the availability of appropriate models for the data source, but a key element in both types of procedures is the grouping, or classification of measurements based on either (i) goodness-of-fit to a postulated model, or (ii) natural groupings (clustering) revealed through analysis. Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters) based on similarity. The clustering problem has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps in exploratory data analysis.  
  
Hadoop

Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then the inputs to the reduce tasks. Typically, both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

Typically, the compute nodes and the storage nodes are the same, that is, the MapReduce framework and the Hadoop Distributed File System are running on the same set of nodes. This configuration allows the framework to effectively schedule tasks on the nodes where data is already present, resulting in very high aggregate bandwidth across the cluster. The MapReduce framework consists of a single master JobTracker and one slave Task Tracker per cluster-node. The master is responsible for scheduling the jobs' component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves execute the tasks as directed by the master.

Minimally, applications specify the input/output locations and supply map and reduce functions via implementations of appropriate interfaces and/or abstractclasses. These, and other job parameters, comprise the job configuration. The Hadoop job client then submits the job (jar/executable, etc.) and configuration to the JobTracker which then assumes the responsibility of distributing the software/configuration to the slaves, scheduling tasks and monitoring them, providing status and diagnostic information to the job-client.In this paper, we implement the k-means++ (with Initial Equidistant Centers) algorithm within the MapReduce framework using Hadoop [10] to make the clustering method applicable to large-scale data. By applying proper <key, value> pairs, the proposed algorithm can be executed in parallel effectively. We conduct comprehensive experiments to evaluate the proposed algorithm. The results demonstrate that our algorithm can effectively deal with large scale datasets.

Litrature Survey  
  
The simplicity and efficiency makes k-means one of the most popular [clustering algorithms](https://www.sciencedirect.com/topics/computer-science/clustering-algorithm). It takes number of cluster k as an input prior clustering. It selects k objects, known as centroids, from the dataset then calculates distance from each object to the centroids using [Euclidean distance](https://www.sciencedirect.com/topics/computer-science/euclidean-distance). The closest objects belong to the centroids create a cluster. Iteratively this process executes till it reaches a finite number of iterations or a [clustering criterion](https://www.sciencedirect.com/topics/computer-science/clustering-criterion) . Traditional K-means is used intensively in several works for [document clustering](https://www.sciencedirect.com/topics/computer-science/document-clustering) and found good result.  
Data clustering algorithms become expensive and slow while dealing with large data repositories.   
To choose the cluster centers smartly, we will compute the K-Mean++ algorithm. K-means++ is just an initialization procedure for K-means. In K-means++ you pick the initial centroids using an algorithm that tries to initialize centroids that are far apart from each other.

In K-Means++ algorithm:

* We choose one data point out of all data points (Xi) on the x-y plane as the cluster center at random.
* For each data point Xi, We compute the distance (di) between Xi and the nearest center that had already been chosen and square that distance (di)2.
* Now, we choose the next cluster center using the weighted probability distribution where a point X is chosen with probability proportional to d(Xi)2. This means the data point is farthest to the first cluster center.
* Repeat Steps 2 and 3 until K centers have been chosen.

Hadoop

Hadoop, an open-source distributed computing framework, revolutionized the processing and analysis of large-scale data when introduced by Yahoo! in 2005. At its core, Hadoop includes the Hadoop Distributed File System (HDFS) for fault-tolerant storage and the MapReduce programming model for parallel data processing. Hadoop's ecosystem has grown to include components like HBase, Pig, Hive, and Spark, expanding its applicability. Scalability and fault tolerance are among its key strengths, making it a choice for major companies like Facebook and Amazon. Nevertheless, Hadoop faces challenges, such as real-time processing and cluster management complexities. Its evolution is ongoing, with efforts to address limitations and adapt to the dynamic field of big data analytics. Hadoop's open-source nature has fostered a thriving community, providing a wealth of resources and support for users. It remains a pivotal player in the big data landscape, even as alternative solutions emerge to address specific use cases and challenges

1.3. Map-Reduce

MapReduce is a programming model and an associated implementation for processing and generating large data sets. The Map and Reduce functions of MapReduce are both defined with respect to data structured as (key, value) pairs. Map takes one pair of data with a type in one data domain, and returns a list of pairs in a different domain. Figure 3 describes the overview of the Map function. The Map function is applied in parallel to every pair in the input dataset. This produces a list of pairs for each call. After that, the MapReduce framework collects all pairs with the same key from all lists and groups them together, creating one group for each key. The Reduce function is then applied in parallel to each group, which in turn produces a collection of values in the same domain. Figure 4 represents the concept of the Reduce function. Each Reduce call typically produces either one value v3 or an empty return, though one call is allowed to return more than one value. The returns of all calls are collected as the desired result list. Thus, the MapReduce framework transforms a list of (key, value) pairs into a list of values. Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system. In general, we use a framework or commodity software called Hadoop (Apache Hadoop) which helps toeasily implement program into map reduce.

Hardware and Software Requirements  
  
Hardware Requirements

Algorithm Used

We implement K-Means (with Initial Equidistant Centers)

on MapReduce using Hadoop platform and run this algorithm on internet

surfing data of static IP address.

The input to the Map function is given into the form of <key, value> pair, where the key

is defined as the cluster center and value is for an object (vector) which

uses to be a cluster. According to the MapReduce discipline, it uses two

files, one that creates clusters with their centroid and others that houses

the objects which are to be clustered . Algorithms for K-Means mapper and Reducer

function are as follows:  
  
**Initialization - Algorithm**

Step 1: start up

Step 2: Input: File having no. of cluster and data points.

Output: Coordination of centriod cluster.

Step 3: For j=0 to L(cluster length)

X\_axis\_centroid=((max\_value\_Xmin\_

value\_X)/(L+1)\*j)+min\_value\_X

Y\_axis\_centroid=((max\_value\_Ymin\_

value\_Y)/(L+1))\*j)+min\_value\_Y

END For.

**Mapper- Function**

Input : Data set\_file

Output : <K,V> pair // K represent the closet center, // V represent the

data point.

Step 1 : if (iteration=first)

Select the centroid through the whole data set.

Otherwise,

Read the reduced based centroid and each

{individual cluster=individual centriod}

Step 2: ED=max\_value //ED- represent Eucledian Distance

Step 3: set index=-1

Step 4: For k=0 to L

Do

D= ED(instance, Centers[K]); // D represent distance

If (D<ED)

Do ED = D && index=i

End for

Step 5: Output <K,V> pair

END  
  
**Reducer- Function**

Input: (K,V) pair // K-represent centroid\_cluster // V-represent data

points various nodes

Output: (K,V) pair //K-represent index of cluster // V-represent data

points value

Step 1: A[i]= datapoints

Step 2: for i=0 to array\_last\_index(L)

Do

Sum the values

END For.

Step 3: Compute the means of datapoint and add it to file

Step 4: if (new\_centroid = previous centroid)

Set converged

Otherwise, go for next iteration

END if

Step 5: Output <K,V> pair

End.