ISSN 2395-1621

Implementation on Performance of Hadoop Using Efficient Data Aware Caching For Big Data

[#1]Amit A. Khodaskar., [#2]Prof.Thombare B.H.

[1]amit_khodaskar@rediffmail.com
[2]babanthombre@gmail.com

[#12]Department of Computer Engineering

Shree Ramchandra College of Engineering, Lonikand, Pune, India.

ABSTRACT

There is a drastic growth of data's in the web applications such data's are said be as Big Data. The Hive queries with the integration of Hadoop are used to generate the report analysis for thousands of datasets. It requires huge amount of time consumption to retrieve those datasets. It lacks in performance analysis. To overcome this problem in medical analysis. A very popular Data Mining Algorithm is used in anywhere. The objective is to store the data persistently along with the past history of the data set and performing the report analysis of those data set. The main aim of this system is to improve performance through parallelization of various operations such as loading the data, index building and evaluating the queries. In this system we analysis different medical problems occurred to the user. We use the medical data set for calculate the how may user have problems. This paper proposes a new method to improve the performance of MapReduce by using distributed memory cache as a high speed access between map tasks and reduce tasks. Map outputs sent to the distributed memory cache can be gotten by reduce tasks as soon as possible. Experiment results show that our prototype's performance is much better than that of the original on small scale clusters. To our knowledge, this is the first effort to accelerate MapReduce with the help of distributed memory cache.

ARTICLE INFO

Article History

Received: 20th July 2017

Received in revised form :

20th July 2017

Accepted: 22nd July 2017

Published online :

24th July 2017

Keywords: Map Reduce, Hadoop, Cache, Distributed Cache

I. INTRODUCTION

The proposed system is implemented using Hadoop for real time processing of the medical data to generate results to establish the hadoop processing. Data are analyzed with Hadoop using MapReduce programming. In smart city Datasets generated by smart homes, smart parking weathser, pollution, Sensex value and vehicle data sets are used for analysis and evaluation. This type of system with full functionality does not currently exist. Similarly, the results demonstrate that the proposed system is more scalable and efficient than existing systems. Moreover, system efficiency is measured in terms of throughput and processing time. MapReduce is a popular computing framework for large-scale data processing. Practical experience shows that inappropriate configurations can result in poor performance of MapReduce jobs, however, it is challenging to pick out a suitable configuration in a short time. Also, current central resource scheduler may cause low resource utilization, and degrade the performance of the cluster.

Big data is an one of the emerging hot research topic because its mostly used in data center application in human society, such as government, climate, finance, and science. Currently, most research work on big data falls in data mining, machine learning, and data analysis. The name itself contains the meaning of data will be so big‖ in large volume of both structured and unstructured data present. These data centers run hundreds of or thousands of servers, so it consumes megawatts of power with massive carbon footprint, and also incur electricity bills of millions of dollars. Data explosion is one of the rising demand for big data processing in recent years and modern data centers that are usually distributed at different geographic regions. efficient.

Hadoop is java based framework that allows to process large data sets in distributed environment. Hadoop has been used by many large scale companies like Amazon, Facebook, and Yahoo. Hadoop consist of two important

concepts: Hadoop Distributed File System (HDFS) and Hadoop Map Reduce. Map Reduce workloads may be very heterogeneous in terms of theirdata size and their re-source requirements , and mixing them within a single instanceof a computing framework may lead to conflicting optimization goals. Therefore,isolating Map Reduce workloads and their data while dynamically balancing theresources across them is very attractive for many organizations .Hadoop is an open source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming mode. It is designedto scale up from single servers to thousands of machines, each offering local computationand storage. Our System relaxes the slot allocation constraint to allow slots to be reallocated to either map or reduce tasks depending on their needs. Second, thespeculative execution can tackle the straggler problem, which has shown to improvethe performance for a single job but at the expense of the cluster efficiency. In viewof this, we propose Speculative Execution Performance Balancing to balance theperformance tradeoff between a single job and a batch of jobs. Third, delay schedulinghas shown to improve the data locality but at the cost of fairness. Alternatively,we propose a technique called Slot PreScheduling that can improve the data localitybut with no impact on fairness. Finally, by combining these techniques together, weform a step-by-step slot allocation system called Dynamic MR that can improve theperformance of Map Reduce workloads substantially.

Goal and Objective:

- Stores large medical database at the same time it can analyze the data mining Algorithm.
- Hadoop processes data fast which is very useful for Real Time System.
- Improves the performance of workloads with maintaining the fairness.

Objectives:

- To minimize the processing
- To reduce the overhead server.

A. MapReduce Framework:

The mapreduce framework consists of two steps namely Map step and reduce step. Master node takes large problem input and slices it into smaller sub problems and distributes these to worker nodes. Worker node may do this again and leads to a multi-level tree structure.

Worker processes smaller problem and hands back to master. In Reduce step Master node takes the answers to the sub problems and combines them in a predefined way to get the output/answer to original problem. The MapReduce framework is fault-tolerant because each node in the cluster is expected to report back periodically with completed work and status updates.

If a node remains silent for longer than the expected interval, a master node makes note and re-assigns the work to other nodes. The detailed MapReduce framework is shown in Fig. 1

B. Hadoop Distributed File System(HDFS):

It is distributed file system designed to run on commodity hardware. This system provides high-throughput access to application data. HDFS is highly fault- tolerant and is designed to be deployed on low-cost hardware. Application that run on HDFS has large data sets. Typically file in HDFS is gigabytes to terabytes in size. It should support tens of millions of files in a single instance. HDFS is designed more for batch process in grather than interactive use by users. Detection of faults and quick, automatic recovery from them is a core goal of HDFS. HDFS has been designed to e easily poratable from one platform to another. HDFS has a Master-slave architecture. An HDFS cluster consist of a single NameNode, a master serves that manages the file system namespaces and regulates access to files by clients. In addition, there are number of datanodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on.

II. LITERATURE SURVEY

Big data is evolving drastically around us. Researchers, scholars defined big data depending on different perspectives. The very common definition of big data is that the datasets that could not be imagined, understood easily, accomplished and processed by conventional information technology software/hardware tools in an expected time. The volume of information increasing every day as people create such large data with the help of communications like voice calls, emails, texts, uploaded pictures, video, and music [8].

MapReduce is used by researchers at Google from 2004. Due to limitless features of big data, researchers understood that a single machine is unable to serve all data computation/analytic solutions, and new environment like distributed system is required to process and store data in parallel [9].

An open source implementation Apache Hadoop similar to MapReduce became available with free of cost for large scale data analytics, big-data applications and other major parallel computations in which large input data is required. Hadoop is adopted by several distinguished and renowned companies like Yahoo!, Facebook and became mainstay [10].

The Hadoop computational model has several distinguished attributed properties. It is simple that its API stipulates a few entry points for the application programmer specified mappers, reducers/combiners, partitioners, for formatting input and output [11].

In addition to basic large scale computational models, lot of software tools is built around as Hadoop ecosystem. These tools are such as Apache Hive, Apache Giraph,

Apache Hama, Apache Mahout all of which harness Hadoop [12].

Since the last ten fifteen years it is observed that Hadoop clusters size is increased, as well as increase in size of RAM memory supported by each machine. The smaller <K, V> size, high amount of data reusability and Hadoop Job interactive ness make it possible to build a robust caching mechanism preferably in -memory for substantial improvement in performance [13].

III. PROPOSED SYSTEM

System architecture:

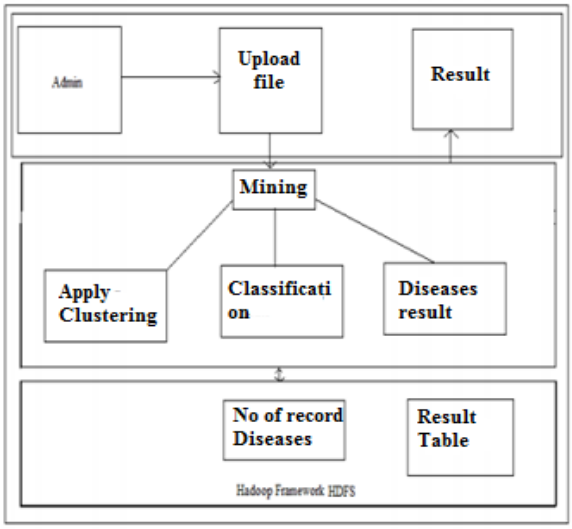

Fig 1. System architecture

Efficient data aware caching system is implemented by using Hadoop incorporated components. Cache manager communicates with task trackers and provides cache items on receiving requests which is implemented in the system. The cache manager uses HDFS, the DFS component of Hadoop, to manage the storage of cache items. In order to access cache items, the mapper and reducer tasks first send requests to the cache manager. Mapper and Reducer classes only accept key value pairs as the inputs which are fixed by Hadoop interface. An open accessed component InputFormat class allows application developers to split the input files of the MapReduce job to multiple file splits and parse data to key value pairs. The component TaskTracker class is responsible for managing tasks, understand file split and bypass the execution of mapper classes entirely. TaskTracker also manages reducer tasks and bypass reducer tasks by utilizing the cached results.

IV. PROPOSED METHODOLOGY

The proposed efficient data aware caching framework is powerful for cache management. The new cache replacement algorithm is implemented and called it as value degree to calculate the value of tuple being replaced. Results show that there is substantial improvement in performance of Hadoop jobs by reducing completion time and storage overhead using efficient data aware caching for big data application.

Module:

Pre-processing Technique : Once data will uploaded then preprocessing done process at well-known Structure.

HDFS Upload: Hadoop Distributed File System (DFS) will be configured for uploading the preprocessed geo data into hadoop. The configuration includes setting VM (hadoop platform) IP and port for connection. FSDataInputStream and FSDataOutputStream is used to upload and download data from hadoop. Different datacenters are analysed for data execution across different datacenters.

User module: File preprocess, after preprocessing user want to clustered, user enter cluster size, after cluster size, user search the records from the catche memory, here we search the data as per user priority, user search the records depending on age, area, gender, months etc.

Graph analysis: In this section we have shown the working of the proposed system. The shows the total files how to use time for searching any records, we check timing for searching the records form existing and proposed system.

Algorithm Steps:

1. Load the File on hadoop framework
2. Preprocess the input file and generate the structure format
3. Apply k-means on the file for clustering the input file
4. Check Weather the File Is available on catche memory or not on hadoop
5. If yes then calculate the result else
6. load the file in the catche memory hadoop
7. Get the result analysis i.e we get the fast input data to the user.
8. Stop

V. RESULT

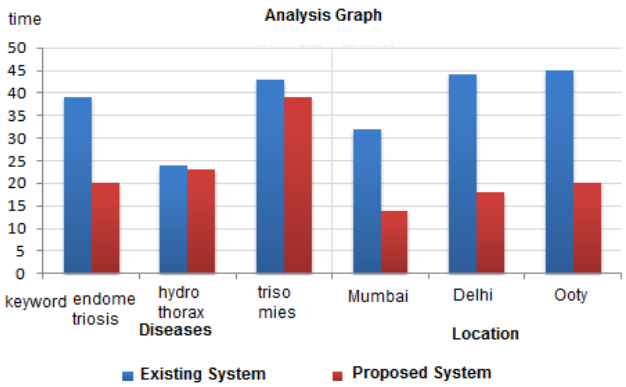

Fig 2. Shows the existing system and proposed system analysis result depending on multiple user search any keyword

VI. CONCLUSION

The proposed efficient data aware caching framework is powerful for cache management. The new cache replacement algorithm is implemented and called it as value degree to calculate the value of tuple being replaced. Results show that there is substantial improvement in performance of Hadoop jobs by reducing completion time and storage overhead using efficient data aware caching for big data application.

REFERENCE

[1] Yaxiong Zhao, Jie Wu, and Cong Liu "Dache: A Data Aware Caching for Big-Data Applications Using the MapReduce Framework" Tsinghua Science and Technology ISSNl l1007-0214l l05/10l lpp39-50 Volume 19, Number 1, February 2014.

[2] Zhu Xudong, Yin Yang, Liu Zhenjun, and Shao Fang "C-Aware: A Cache Management Algorithm Considering Cache Media Access Characteristic in Cloud Computing", Research Article, Hindawi Publishing Corporation, Mathematical Problems in Engineering, Article ID 867167, 13 pages, Volume 2013.

[3] Meenakshi Shrivastava, Dr. Hans-Peter Bischof "Hadoop-Collaborative Caching in Real Time HDFS" Computer Science, Rochester Institute of Technology, Rochester, NY, USA.

[4] Dachuan Huang, Yang Song, Ramani Routray, Feng Qin "SmartCache: An Optimized MapReduce Implementation of Frequent Itemset Mining" The Ohio State University, IBM Research – Almaden.

[5] Yingyi Bu, Bill Howe, Magdalena Balazinska, Michael D. Ernst "HaLoop: Efficient Iterative Data Processing on Large Clusters" Department of Computer Science and Engineering University of Washington, Seattle, WA, U.S.A. 36th International Conference on Very Large Data Bases, September 1317, 2010, Singapore.

[6] Ganesh Ananthanarayanan, Ali Ghodsi, AndrewWang, Dhruba Borthakur, Srikanth Kandula, Scott Shenker, Ion Stoica "PACMan: Coordinated Memory Caching for Parallel Jobs" University of California, Berkeley, Facebook, Microsoft Research, KTH/Sweden.

[7] Gurmeet Singh, Puneet Chandra and Rashid Tahir "A Dynamic Caching Mechnism for Hadoop using Memcached" Department of Computer Science, University of Illinois at Urbana Champaign.

[8] Executive Office of the President " Big Data: Seizing Opportunities, Preserving Values" May 2014.

[9] [14] Min Chen, Shiwen Mao, Yunhao Liu "Big Data: A Survey" Published online: 22 January 2014, Springer Science+Business Media New York 2014.

[10] Avraham Shinnar, David Cunningham, Benjamin Herta, Vijay Saraswat "M3R: Increased Performance for InMemory Hadoop Jobs", roceedings of the VLDB Endowment, Vol. 5, No. 12,38th International Conference on Very Large Data Bases, Istanbul, Turkey, August 27th 31st 2012.

[11] Tom White "Hadoop: The Definitive Guide",Third edition,Oreilly, ISBN: 978-1-449-31152-0.

[12] Venkatesh Nandakumar "Transparent in-memory cache for Hadoop-MapReduce" A thesis submitted in conformity with the requirements for the degree of Master of Applied Science Graduate Department of Electrical and Computer Engineering University of Toronto, 2014.

