Study and Analysis of Big data with MapReduce Framework

K. Rama Krishna Reddy, B.G. Obula Reddy

Abstract— Exponential growth in data has been observed in recent years. This huge amount of data has caused a new kind of problem. Existing RDBMS systems cannot handle large data or they are not effective in managing them. Major Big Data problems are storage and handling. Hadoop is displayed in storage and processing solutions in the form of HDFS (Hadoop Distributed File System) and MapReduce. Traditional systems are not intended for Big Data processing, and they can also process structured data. The financial sector is one of the challenges in Big Data. In this work, unstructured data is processed by Hadoop MapReduce. An effective processing of unstructured data is analyzed and explained.

Keywords: Big data, Hadoop, HDFS, MapReduce.

1. INTRODUCTION

We live in a date of age. It is difficult to calculate the total amount of data electronically stored, but the IDC calculates the size of the "automated world" 4.4 Zetta Bytes in 2013, and estimates the development ten times by 2020, up to 44 ZB. Approximately one thousand EB, one million BP or one billion TB. This is more than a circle of unity for all who live on the planet. So there are a lot of data that indicates that they have an impact. Most data is protected by larger web properties (such as web crawlers) or logical or money-related parameters, would you say no? Do large amounts of data affect associations or smaller people? The model is that the data on each of them has evolved, although perhaps more importantly, to the extent that the data produced by machinery produced as part of the Internet of Things is significantly higher than that produced by individuals. Machine files, RFID files, sensor systems, vehicle GPS tracking, local exchange for data exchange. Freely available amount of data is also created annually. Associations should never process your data. The result will be largely motivated by its ability to generate incentives from other associations. Customers who need to store and retrieve data store information for a long time in the database and process them using SQL queries. The web has changed most of the suspicions of this time. The information on the Internet is unstructured and essential, and databases cannot save or increase the information in the scheme to store and process them. Google was one of the most important associations to solve the problem. To find questions, she had to download and index the entire Internet. They created the basis for extensive information management, derived from the options "map" and "reduced".

Revised Version Manuscript Received on 22 February, 2019.

K. Rama Krishna Reddy, Associate Professor, Department of CSE, Malla Reddy Engineering College (A), Telangana, India

B.G. Obula Reddy, Associate Professor, Department of CSE, Malla Reddy Engineering College (A), Telangana, India

2. LITERATURE REVIEW

Literature review described as the summary or re-organising the related information from different sources to understand the research problem. In this review how data analysis done for big data through data mining techniques is studied.

[1] MapReduce is a planning model and appropriate implementation for the management and production of complete data sets. Customers set up a card task where the main / calculation procedures are combined to form half the main route / cluster table and reduce the work of trade unions that have all the values associated with the central key. This model shows a large number of paper-based companies. Projects written in this useful style are parameterized in parallel and executed in a naturally large group of production machines. The enforcement system covers the fine elements of the Information Data Unit, reserving the program execution in machine mode, solving machine frustration and addressing the compliance of the required machines. This allows software engineers who are not involved in parallel and distributed infrastructures to effortlessly use assets that are significantly dispersed. Our use of MapReduce is still applicable to large-scale machine tools, and is very versatile: the typical MapReduce calculation makes up countless terabytes of data in a large number of machines. Developers believe that the system is easy to use: several MapReduce programs have been updated, and more than a thousand MapReduce tasks work consistently in Google's clusters.

[2] In fact, programming systems often operated without creating a direct expressive model. This can lead to complex problems that can ruin further development, which is relatively unavoidable, since in the best case, the most important product documentation is the source code. To solve this problem, learn how he focused on software modeling inductance using automatic recording learning algorithms. In both cases, records generated using a good scheduling system can be significant and the calculation of the derivative may exceed the processing limit of one computer. This document provides a generic and flexible way to obtain behavioral patterns capable of handling critical performance records using commonly used and parallel algorithms using the MapReduce programming model and executing a group of connected execution centers. The approach consists of two divided phases, which destroy and combine the model. The calculation for each phase was done using MapReduce. With MapReduce's parallel data boundary, the problem of

removing major registry behavior patterns can be solved. This strategy is updated by Hadoop. Analyzes in the Amazon clusters indicate our approach to productivity and adaptability.

[3] The Google MapReduce programming model is used to handle large datasets in parallel. We provide the first exact model description, including the Google-specific Sawzall domain's progress. For this purpose, we create Primary Documents in MapReduce and Sawzall and record our results as executable specifications. We also identify and resolve some of the errors in the relevant documents in the informal presentation. We use the recorded functional program (especially Haskell) as a tool for design restoration and executable specification.

[4] MapReduce is the general basis for cluster-based distributed batch processing. In order to streamline debt recovery capabilities, many MapReduce performances are shown in the overall performance of each card and reduce allocations before they are consumed. In this document, we offer an appropriate MapReduce technique for transferring data between administrators. This extends the MapReduce programming model beyond cluster management and can reduce processing time and improve group activity in group activities. We introduced a custom Hadoop MapReduce structure that provides total online sharing so customers can see the "first respond" of the treated call. Our Hadoop Online Prototype (HOP) also supports continuous queries that allow aggregating MapReduce projects for things like incident observation and energy management. Bounce retains Hadoop's non-critical malformation properties and can run MapReduce programs that feature an unmodified client.

[5] In recent years, in the rapid development of data, Industries and the rest of the world, the academic world needed a smart data retrieval device that would be useful in removing huge amounts of data. The MapReduce system is primarily designed to record scaled data applications to support vital management. Since its launch, incredible research has been done to make this process more natural for customers. This allows them to intensify the execution of large applications based on data. Our homepage highlights the key role of improving the efficiency of various applications that use the latest MapReduce models, as well as large scale data set processing. A comparative study of Dot models is compared to Apache Hadoop, and Phoenix will be a fundamental discussion of performance time and adaptation to internal failures. Finally, the call for updating the MapReduce calculation in a given region will be unusual, for example, recalculation, harmonized processing of issues, Métis database, etc.

3. PROBLEM STATEMENT

Traditional database systems are not intended to handle large volumes of data that we have known over the years. It is also expensive to increase the processing power of these systems. In addition, traditional systems can process only structured data. Unstructured data is the most important part of the data collected over the last two years. The existing system works on the same server, making vertical growth more complex and expensive.

Limitations

The two main drawbacks are:

1. Large data cannot be stored and processed
2. Unstructured data cannot be processed

It is limited how this system can be made vertical. Most of these individual servers are high-end or modified, so they are not profitable.

4. RECOMMENDED SYSTEM

The traditional RDBMS is not designed to handle unstructured data, and the size of the data it can handle is limited. The biggest problems with data processing are storage and processing. In this work, the Hadoop Framework is used to address storage and processing problems. Hadoop stores data in a cluster that is distributed over the network and processes it. Solve the problem by increasing storage and processing power. This can easily be achieved by increasing the number of nodes in the cluster. Data storage as well as processing distributed in cluster nodes significantly reduces the processing time of Big Data. It also eliminates the need for high-quality or custom-made equipment, which is very expensive.

5. CONCLUSION

In this paperwe describes how unstructured data can be efficiently processed using the MapReduce programming model. The Hadoop Framework is a solution to big data-processing problems. In this work, a series of unstructured inventory data is used to demonstrate the implementation of MapReduce tasks. In addition, the relevant sections are shown. This work provides a better understanding of the implementation of the MapReduce tasks by unstructured registry data and encourages further research to find more efficient ways of processing non-structured registry data.

REFERENCES

1. L. Augusteijn. Sorting morphisms. In S. Swierstra, P. Henriques, and J. Oliveira, editors, 3rd International Summer School on Advanced Functional Programming, volume 1608 of LNCS, pages 1–27. Springer-Verlag, Sept. 1998.
2. J.W. Backus. Can Programming Be Liberated From the von Neumann Style? A Functional Style and its Algebra of Programs. Communications of the ACM, 21(8):613–641, 1978.
3. R. Bird and O. de Moor. Algebra of programming. Prentice-Hall, Inc., 1996.
4. R. S. Bird. An introduction to the theory of lists. In Proceedings of the NATO Advanced Study Institute on Logic of programming and calculi of discrete design, pages 5–42. Springer-Verlag, 1987.
5. G. E. Blelloch. Programming parallel algorithms. Communications of the ACM, 39(3):85–97, 1996.
6. A. Borodin and J. E. Hopcroft. Routing, merging and sorting on parallel models of computation. In STOC'82: Proceedings of the fourteenth annual ACM symposium on Theory of computing, pages 338–344. ACM Press, 1982.
7. L. Boug´e, P. Fraigniaud, A. Mignotte, and Y. Robert, editors. Proceedings of the 2nd International Euro-Par Conference on Parallel Processing, 2 volumes, EURO-PAR'96, volume 1123–1124 of LNCS. Springer-Verlag, 1996.
8. W.-N. Chin, J. Darlington, and Y. Guo. Parallelizing conditional recurrences. In Boug´e et al. [7], pages 579–586. Volume 1/2.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

9. P. Zadrozny and R. Kodali, Big Data Analytics using Splunk, Berkeley, CA, USA: Apress, 2013.
10. F. Ohlhorst, Big Data Analytics: Turning Big Data into Big Money, Hoboken, N.J, USA: Wiley, 2013.
11. J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Commun ACM, 51(1), pp. 107-113, 2008.
12. F. Li, B. C. Ooi, M. T. Özsu and S. Wu, "Distributed data management using MapReduce," ACM Computing Surveys, 46(3), pp. 1-42, 2014.
13. C. Doulkeridis and K. Nørvåg, "A survey of large-scale analytical query processing in MapReduce," The VLDB Journal, pp. 1-26, 2013.
14. S. Sakr, A. Liu and A. Fayoumi, "The family of mapreduce and large-scale data processing systems," ACM Computing Surveys, 46(1), pp. 1-44, 2013.
15. The emergence of “big data” technology and analytics Bernice Purcell –Holy Family University.
16. The Forrester Wave™: Big Data Predictive Analytics Solutions, Q1 2013 by Mike Gualtieri, January 3, 2013