International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT217689

Optimal Common Job Block Table (CJBT) to improve the Performance in Hadoop framework

Pinjari Vali Basha
M. Tech Scholar, Computer Science and Engineering, JNTUA College of Engineering, Ananthapuramu, Andhra Pradesh, India

Article Info
Volume 7, Issue 6
Page Number : 346-350

Publication Issue :
November-December-2021

Article History
Accepted : 12 Dec 2021
Published : 26 Dec 2021

ABSTRACT

By rapid transformation of technology, huge amount of data (structured data and Un Structured data) is generated every day. With the aid of 5G technology and IoT the data generated and processed every day is very large. If we dig deeper the data generated approximately 2.5 quintillion bytes.

This data (Big Data) is stored and processed with the help of Hadoop framework. Hadoop framework has two phases for storing and retrieve the data in the network.

- Hadoop Distributed file System (HDFS)
- Map Reduce algorithm

In the native Hadoop framework, there are some limitations for Map Reduce algorithm. If the same job is repeated again then we have to wait for the results to carry out all the steps in the native Hadoop. This led to wastage of time, resources. If we improve the capabilities of Name node i.e., maintain Common Job Block Table (CJBT) at Name node will improve the performance. By employing Common Job Block Table will improve the performance by compromising the cost to maintain Common Job Block Table.

Common Job Block Table contains the meta data of files which are repeated again. This will avoid re computations, a smaller number of computations, resource saving and faster processing. The size of Common Job Block Table will keep on increasing, there should be some limit on the size of the table by employing algorithm to keep track of the jobs. The optimal Common Job Block table is derived by employing optimal algorithm at Name node.

Keywords : Common Job Block Table, Least Recently Used, Improved Hadoop.

I. INTRODUCTION

Using Hadoop framework, it is very efficient to handle big data storage as well as its processing. Hadoop uses large clusters of commodity hardware to store and process big data in a distributed fashion. Open Source, Massive data storage and faster processing capabilities made it very popular.

To retrieve the data from the distributed environment requires lot of computations. More efficient algorithms are needed to handle such cases. Map Reduce framework is a efficient algorithm to process the huge data sets. Even Map Reduce algorithm is efficient to process the big data it has some limitations.

- Task scheduling is based on the location of the data.
- After finishing all mapping, reducing can be started.
- Intermediate data generated during map reduce process is destroyed after use.
- To provide data location and resource allocation it requires lot of efforts.
- Map reduce treats each job as a new job and does all the computations again.

In Hadoop framework there are Three daemons. Name node, Secondary Name node and Data node. Name node holds the meta data of file that is distributed in the cluster, Secondary Name node holds the replica of meta data of file, that is used in case of master Name node failure and Data node holds the actual data of file that is divided into blocks.

Each block has three replicas in the cluster. When there is a request for accessing the file from the client, the request send to Name node then name node will reply the meta data of file to the client, client will convert that meta data in the form of HDFS and then the request would be sent to the Data nodes which are in the cluster.

Map reduce algorithm is implemented at each Data node. Map algorithm will find the data sets present at each Data node. Reduce algorithm will aggregate all the data sets (blocks) into one file and then the file will be transferred to the client machine.

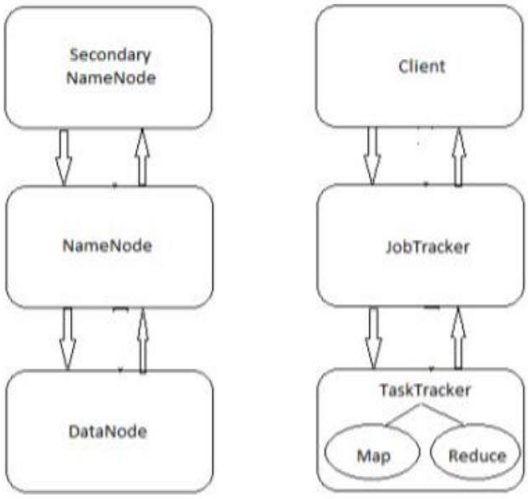

Fig.1. Hadoop Daemons and Interactions

The above diagram will illustrate how Hadoop Daemons will interact in the process of implementing the request from the client machine.
Each request from the client is treated as a new request so that there is wastage of Time and Resources. The improved Hadoop will illustrate how this problem is minimized.

II. IMPROVED HADOOP

As discussed above, In the native Hadoop framework every request is treated as new request. When there is a request for the same file again, then it is treated as a new request then all the steps are repeated again, this leads to wastage of time and resources. In the improved Hadoop we improve the capabilities of Name node by employing a special table called Common Job Block Table (CJBT). CJBT holds the data of the data of files, which act as Cache for the files. When there is a request from the client, The Name node first inspect the CJBT. If the file information found in the CJBT then the Name node gets the file directly from the nodes which is already computed in the previous request. If the file information is not available in the CJBT, then it is treated as a new request.

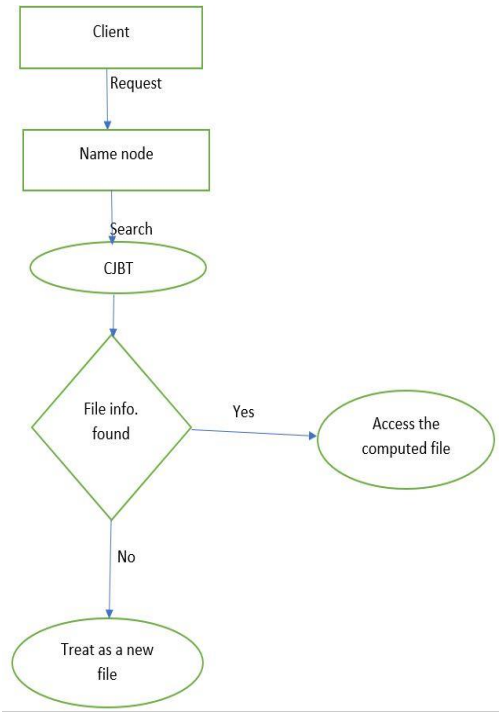

Fig.2. Flow chart for Improved Hadoop

III. IMPLEMENTATION

The implementation of Common Job Block Table at Name node will improve the performance by minimizing the time to access the file and computation power and resources by compromising the cost to implement Common Job Block Table. Common Job Block Table act as a cache for the files which contains the following attributes.

- Common job name
- Common feature
- Block name

Common Job name	Common feature	Block name
Bigdata.txt	Xxxxyyyyzzzz	B1, B2
Bigdata1.txt	Xxxxxyyyyzzz	B1, B3
Bigdata2.txt	Xxxx	B3, B4
Bigdata3.txt	Xxxxyyyy	B2, B3

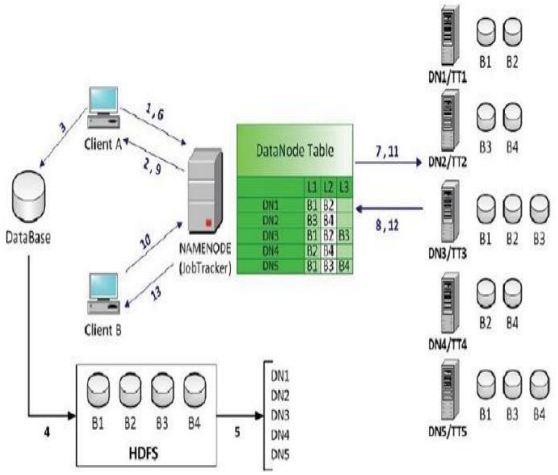

Fig.3. Native Hadoop framework

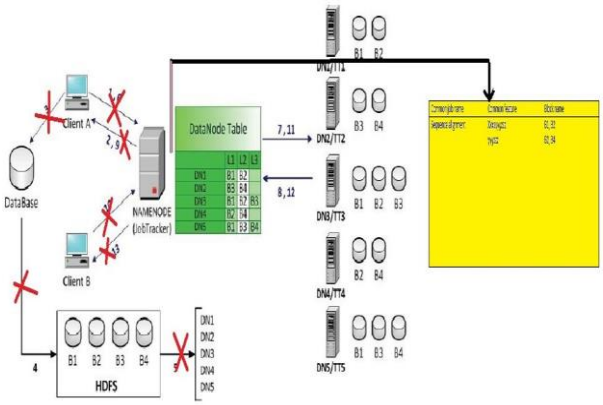

Fig.4. Improved Hadoop framework

The size of Common Job Block Table keeps on increasing in real time. There should be some limit on the size of CJBT. To get optimal size of the CJBT we implement optimal algorithm on the Common Job Block Table. Least Recently Used (LRU) algorithm will give the best results. We replace the existing file in the CJBT with the new file when it is not used in recent.

IV. EXPERIMENTAL RESULTS

The results are compared between native Hadoop framework and Improved Hadoop framework. If the file is not processed previously then only Map Reduce task is performed. After Map Reduce tasks, results are stored at Data Nodes and an entry is made at CJBT. If the same file is requested by the client again, then it searches first in the CJBT to get optimal performance.

By this the recompilations are reduced and data transfer within the network is reduced. Data Nodes required during the action is very less, it further helped to reduce the energy as well.

File name	Time required for Map Reduce computations (in seconds)	Time to show output on screen (in seconds)
Bigdata1.txt	20	21
Bigdata2.txt	25	27
Bigdata3.txt	18	20

Table.1. Native Hadoop results

File name	Time to show output on screen (in seconds)	Time to show output on screen in improved Hadoop (in seconds)
Bigdata1.txt	21	5
Bigdata2.txt	27	6
Bigdata3.txt	20	3

Table.2. Improved Hadoop results

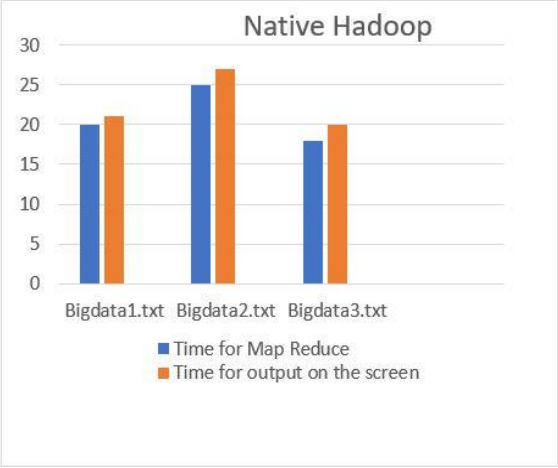

Fig.4. Native Hadoop graph results

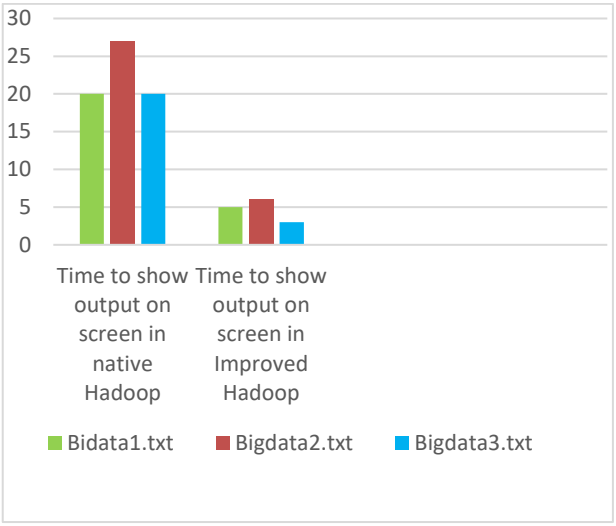

Fig.5. Improved Hadoop graph results

V. CONCLUSION

By enhancing the capabilities of Name node, we have enhanced the performance of Hadoop framework. By the implementation of Common Job Block Table (CJBT) at Name node will reduce the time to access the file and resource minimization by compromising the cost to implement Common Job Block table at Name node. And also implement Least Recently Used algorithm to put cap on the size of the Common Job Block Table.

VI. REFERENCES

[1]. Sachin Arun Thanekar, K. Subrahmanyam, A. B. Bagwan, “Big Data and MapReduce Challenges, Opportunities and Trends”, International Journal of Electrical and Computer Engineering (IJECE) Vol. 6, No. 6, pp. 2911~2919, December 2016.

[2]. Sachin Arun Thanekar, K. Subrahmanyam, A. B. Bagwan, “A Study on Digital Forensics in Hadoop”, I J C T A, 9(18), pp. 8927-8933, 2016.

[3]. H. Alshammari; J. Lee; H. Bajwa, "H2Hadoop: Improving Hadoop Performance using the Metadata of Related Jobs," in IEEE Transactions on Cloud Computing , vol.PP, no.99, pp.1-1

[4]. H. Alshammari, J. Lee and H. Bajwa, "Evaluate H2Hadoop and Amazon EMR performances by processing MR jobs in text data sets," 2016 IEEE

Long Island Systems, Applications and Technology Conference (LISAT), Farmingdale, NY, 2016, pp. 1-6.

[5]. Ibrahim Abaker Targio Hashem, Ibrar Yaqoob, Nor Badrul Anuar, Salimah Mokhtar, Abdullah Gani, Samee Ulah Khan, "The rise of "big data" on cloud computing : Review and open research issues ", Elsevier Information Systems 47 (2015) 98–115.

[6]. Lidong Wang1 and Chery Ann Alexander, " Big Data: Infrastructure, technology progress and challenges", Journal of Data Manaagement and Computer Science Vol. 2(1), pp. 001-006, July, 2015. 7Wei Fan, Albert Bifet, "Mining Big Data: Current Status, and Forecast to the Future", SIGKDD Explorations Volume 14, Issue 2, 2012.

[7]. H. Alshammari H. Bajwa L. Jeongkyu "Enhancing performance of Hadoop and MapReduce for scientificdata using NoSQL database" , Systems Applications and Technology Conference (LISAT) 2015 IEEE Long Island, 2015.

[8]. Z. Asad, M. Asad Rehman Chaudhry and D. Malone, "CodHoop: A system for optimizing big data processing," 2015 Annual IEEE Systems Conference (SysCon) Proceedings, Vancouver, BC, 2015, pp. 295-300.

[9]. Zakia Asad, Mohammad Asad Rehman Chaudhry, "A Two-Way Street: Green Big Data Processing for a Greener Smart Grid", Systems Journal IEEE, vol. 11, pp. 784-795, 2017, ISSN 1932-8184.

[10]. Zakia Asad, Mohammad Asad Rehman Chaudhry, David Malone, "Greener Data Exchange in the Cloud: A Coding-Based Optimization for Big Data Processing", Selected Areas in Communications IEEE Journal on, vol. 34, pp. 1360-1377.

[11]. Zakia Asad, Mohammad Asad Rehman Chaudhry, "A Set Cover Based Efficient Solution for the Complementary Index Coding Problem", Ubiquitous Wireless Broadband (ICUWB) 2015 IEEE International Conference on, pp. 1-5, 2015.

[12]. Abdelrahman Elsayed, Osama Ismail, and Mohamed E. El-Sharkawi, "Map Reduce: State-of-the-Art and Research Directions", International Journal of Computer and Electrical Engineering, Vol. 6, No. 1, February 2014.

[13]. K. Grolinger, M. Hayes, W. Higashino, A. L'Heureux, D. S. Allison, M. A. M. Capretz, "Challenges for MapReduce in Big Data", IEEE 10th 2014 World Congress on Services (SERVICES 2014), June 27-July 2, 1014, Alaska, USA.

[14]. Nilam Kadale, U. A. Mande, "Survey of Task Scheduling Method for Map Reduce Framework in Hadoop", International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868 Foundation of Computer Science FCS, New York, USA 2nd National Conference on Innovative Paradigms in Engineering & Technology (NCIPET2013).

[15]. Diana Moise, Thi-Thu-Lan Trieu, Gabriel Antoniu, Luc Boug_e "Optimizing Intermediate Data Management in MapReduce Coputations". CloudCP 2011 { 1st International Workshop on Cloud Computing Platforms, Held in conjunction with the ACM SIGOPS Eurosys 11 conference, Apr 2011, Salzburg, Austria. 2011.

[16]. Shafali Agarwal, Zeba Khanam, " MapReduce: A Survey Paper on Recent Expansion", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 6, No. 8, 2015.

Cite this article as :

Pinjari Vali Basha , "Optimal Common Job Block Table (CJBT) to improve the Performance in Hadoop framework", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 7 Issue 6, pp. 346-350, November-December 2021. Available at

doi : https://doi.org/10.32628/CSEIT217689

Journal URL : https://ijsrcseit.com/CSEIT217689