



VARIOUS ISSUES IN HADOOP DISTRIBUTED FILE SYSTEM, MAP REDUCE AND FUTURE RESEARCH DIRECTIONS

Dr.K.Umapavankumar¹, Dr.SVN. Srinivasu²,
Dr.S.N.Tirumala Rao³, VVAS Lakshmi⁴,

¹Associate Professor,CSE Department
Narasaraopeta Engineering College
Dr.kethavarapu@gmail.com

²Professor,CSE Department
Narasaraopeta Engineering College
drsvnsrinivasu@gmail.com

³Professor & HOD, CSE Department
Narasaraopeta Engineering College
nagatirumalarao@gmail.com

⁴AssistantProfessor,CSE Department
Narasaraopeta Institute of Technology
vvaslakshmi@gmail.com

June 24, 2018

Abstract

The current usage of internet and various apps populating huge amounts of data, the mandate to store this data to make use of this to various analytics. The analytics may involve click stream analysis, sentiment analysis and various recommendations to the customers. The ultimate goal is to reach the customers by analysing

the profiles. The main issue in the current market is how to store the huge amounts of the data and how to analyse the meaningful insights from the data. The answer is Hadoop Distributed File System (HDFS), Map Reduce and Machine Learning (ML). The current article dealing with the architectural aspects of the above mentioned along with issues existing. The article also floats some of the research directions and gaps existing in the HDFS and MR. The importance of HDFS are to store huge amounts of the data no matter whether the data is Structured, Semi-Structured and Unstructured formats. The HDFS configurations along with purpose were discussed in the article. Map Reduce is parallel and distributed paradigm where it process huge amounts of the data. The simple analytics can be shared with the tools Like Hive and Pig Latin. The belief is that the article gives the conceptual aspects of HDFS, MR along with issues existing in the process of File storage and processing. The possible ways of extending the storage logics and processing logics were also touched.

Keywords:HDFS, MR, Hive, Pig, Parallel and Distributed.

1 Introduction

Hadoop is a frame work to handle the huge data storage and processing the data with parallel and distributed perspective. The HDFS provides a versatile storage for various kinds of the data such as RDBMS (MySQL, Oracle etc.), Java Script Object Notation (JSON) and XML. There is a provision of handling unstructured data processing such as audio, video and images through HDFS and Map Reduce. The development of Map Reduce code and the kind of API usage is dependent on both Java API and Hadoop API. Hadoop provides a separate environment so as to handle the storage and processing logic. The organization of the paper is in section I the architectural aspects of HDFS and Map Reduce were discussed. Section II describes the issues existing in the storage and processing aspects of the system. In Section III the possible research direction in the improvement of HDFS and MR paradigms. Section IV describes the conclusion of

the article.

2 Architectural aspects of HDFS and MR

Apache Hadoop provides a frame work to handle the storage and processing logic to handle huge amounts of the data. The data itself becomes the problem in the data processing then we can call that as a big data. The big data term refers to the Volume, Velocity, Variety and sometimes veracity. Here the mention of some of the big data scenarios is a must. The medical data, the sports data and sensor data of environment need huge repositories to store. An airline jet collects 10 TB of sensor data for every 30 minutes of flying time.

NYSE generates about 1TB of new trade data per day to Perform stock trading analytics to determine trends for optimal trade. As of 2011, there are 500,000,000 active Facebook users. Approximately 1 in Every 13 People on earth, half of them are logged in on any given day. A record-breaking 750 Million Photos were uploaded to Facebook over New Years weekend. There are 206.2 million internet users in the US. That means 71.2% of the US web audience is on Facebook. Facebook users spend 10.5 billion minutes (almost 20,000 years) online on the social network. Facebook has an average of 3.2 Billion Likes and comments are posted every day.

The number of servers worldwide will grow by 10X.Amount of information managed by world data centers will grow by 50X.Number of "files" enterprise data centers handle will grow by 75X. The exciting feature of Hadoop usage is the availability of various distributions in the market. The example Hadoop distributions involve MapR Technologies, Hortonworks, Cloudera, IBM, AWS, Teradata, Intel, Microsoft, and Pivotal Software

The focus is on Apache Hadoop and the configuration of the Hadoop Cluster involves the following XML files such as core-site.xml, mapred-site.xml, hdfs-site.xml, Hadoopenv. sh. The files configurations are mentioned.

Core-site.xml

```

< property >
< name > hadoop.tmp.dir < /name >
< value > /home/uma/data < /value >
< /property >
< property >
< name > fs.default.name < /name >
< value > hdfs : //localhost : 6789 < /value >
< /property >

```

mapred-site.xml

```

< property >
< name > mapred.job.tracker < /name >
< value > localhost : 1234 < /value >
< /property >

```

Hdfs-site.xml

```

< property >
< name > dfs.replication < /name >
< value > 1 < /value >
< /property >

```

While configuring Hadoop cluster we have to setup the above mentioned xml files. The Hadoop architecture involves Hadoop 1.x and Hadoop 2.x. In Hadoop 1.x the daemons are like Name Node, Data Node, Job Tracker, Task Tracker and Secondary Name Node. The block size in Hadoop 1.x is 64 MB, Hadoop follows the Block size based allocation in the HDFS storage which is taken from Google File System.

Name Node is a Master daemon comes to HDFS which is responsible to take care about the Meta data and File system aspects of HDFS. The Data Node belongs to Slave and comes under HDFS and follows the instructions of Name Node and stores the actual data. The Job Tracker is a Master daemon and comes under MR, responsible to split the job into tasks and assign the tasks to Task Tracker. The Task Tracker is a Slave daemon comes under MR and running of the actual task based on the Data Node. The Secondary Name Node is a daemon belongs to Master and comes under HDFS which is backup copy of Name Node.

For example in HDFS a file of 240MB is getting stored means the requirement is 1 block stores 64 MB the default replication factor is 3. The number of blocks required to store 240MB file is 4 blocks and total blocks required is $3 * 4 \text{ blocks} = 12 \text{ blocks}$. In case of MR the file from HDFS is accessed by the Task Tracker based on the Data Node and the process of reading requires 4 blocks only. In case of writing a file total of 12 blocks are required.

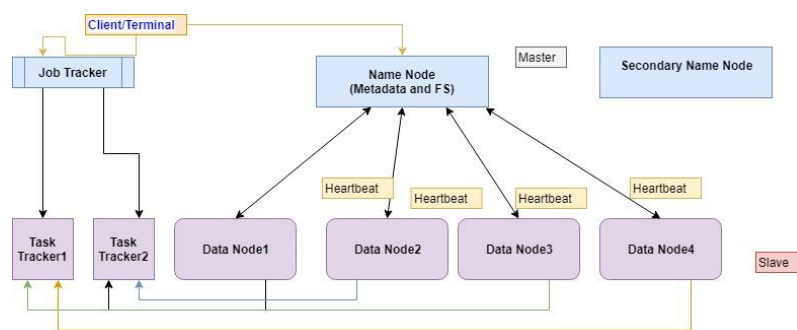


Figure 1: HDFS and MR architecture (Hadoop 1.x)

3 Issues in HDFS and MR

The Figure 1 depicts the abstract flow of the data in HDFS and MR, as mentioned earlier the issue in HDFS storage is block based storage which requires more blocks in the allocation of memory rather than exact requirement.

The default replication factor in Hadoop environment is 3 so every time the same file need to be replicated thrice, which leads to the bulk memory usage even for the normal files which are not that much important.

The another issue in the HDFS architecture is with heart beat update from each and every data node to the name node which consumes more bandwidth and network usage in case of heavy data processing.

The most important issue in the processing of the data with MR is individual task tracker monitoring which consumes more resources and establishing a proxy task tracker without confirmation consumes more time and unnecessary job processing.

The important point observed in the HDFS storage is handling of the small files (of course Hadoop is meant for Huge data) is taking more time is required to store and access these files. Issue here is there is no solution or mechanism in Hadoop to handle the small files processing with in the stipulated time or at least less than the time required to handle huge files.

The Schedulers available in Hadoop are Capacity Scheduler, FIFO Scheduler, Fair Scheduler and Performance Fairness Scheduler. All these schedulers are dependent on either file size or priority of the file. The cases like part time schedulers helps us to invoke a job scheduling and running of the jobs based on the common file or properties required in the data processing.

The Reducer Skew is other issue in the running of MR tasks. For example the implementation of wiki page counts statistics through online mode may be a chance of searching from a particular web site is more. In that case only a specific reducer is taking care about all the Mappers which are involve in the corresponding search engine.

The input split follows sequential input format in the processing of unstructured data like audio, video and image kind of the data, the existing approach follows the encryption based logic to handle the (Key, Value) pair data processing, which consumes more time and requires complex data processing.

The major issue in the hadoop architecture is single point of failure, in case of hadoop 1.x there is secondary name node which is not a hot back up of name node which leads to loss of meta data and file system image of the name node. In case of Hadoop 2.x there exist passive name node which records the response and Metadata information form the data nodes but could not allowed to give the instructions to the Data Nodes, leads to the extra resource consumption in the architecture.

The tool Hive is meant for handling data base kind of activities with the help of HiveQL. The issue with the Hive tool is there is a need to establish a derby data base in the configuration of hadoop cluster. Hive can be instantiated from HDFS and local file system but there is a need of temporary data base to handle the activities of Hive.

4 Research Directions to the Issues in HDFS and MR

HDFS storage is block based storage which requires more blocks in the allocation of memory rather than exact requirement. The block based storage consumes more memory rather than required. The proposed solution is variable length block size mechanism in the allocation of the memory to a file. The principle here is based on the required file size Hadoop could allocate the block which is near to the leftover size of the file.

For example the file size is 440 MB, if the existing block allocation is followed based on 128 MB, then we need 4 blocks each 128 MB i.e., 512 MB out of 440 MB only utilized and 72 MB is a waste. With the proposed method of varying length block size memory allocation first 3 blocks can be allocated with 128 MB i.e., 384 MB allocated the remaining 56 MB is allocated with variable length block size 64 MB so the wastage is reduced and it is only 8 MB.

Table 1: Default Allocation Blocks vs. Proposed Allocation of Blocks

File Parameters	Existing Allocation	Proposed Allocation	Remarks
Required Size: 440 MB	Allocated Size: 512 MB (Fixed Block)	Allocated Size: 384 MB + 64 MB (Variable Block)	Memory Wastage: 72 MB
Required Blocks: 4	Required Blocks: 4	Required Blocks: 3 (Fixed) + 1 (Variable)	Memory Wastage: 8 MB

The method of Variable Length Block Allocation (VLBA) greatly reduces the memory wastage when compared with the existing method of block allocation in Hadoop. The graph shows the difference between the allocations of sizes with existing and proposed method. The proposed method outperforms when compared with the existing the decreased percentage of memory with the proposed method is 88.88% which is very good improvement in the memory allocation strategy.

The architectural model of Hadoop follows the replication factor of 3 so the same file must be copied 3 times in the various data nodes. The scenarios like the credit card data with the details of current customers and the customers who disposed their cards of a particular bank. Obviously the details of the left over customers

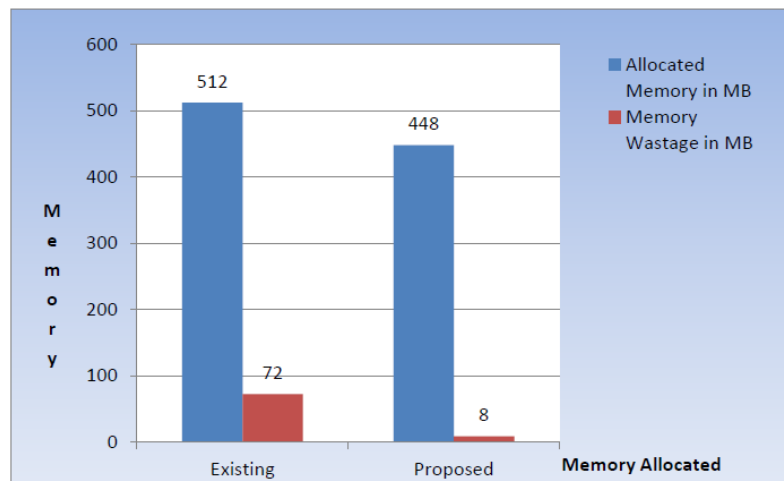


Figure 2: Existing vs. proposed VLBA Method of Block Allocation.

need not be replicated thrice and only the current customers need to replicate thrice.

Assume that in huge data storage the replication causes lot of memory wastage and usage of resources in Name Node metadata and Data Node storage. The proposed method of File Context Based Replication (FCBR) identifies the context of the file with the help of domain experts and based on the kind of the data existing in the files and importance of the data the replication remains same for the sensitive data whereas the files which contain the data for documentation purpose need to have the replication of only 1.

For example the file size is 500 GB and as per the FCBR it contains the sensitive data, so we need to follow the replication of 3 then the requirement is 1500 GB i.e., 1.5 terabytes. In the same context the file size 900 GB and as per the FCBR it contains the document purpose data, so with the existing method of replication Hadoop allocates $3 \times 900 \text{ GB} = 2.7 \text{ TB}$ but with the proposed method of FCBR it is only 900 GB. IV. Conclusion Hadoop provides a framework to handle bulk data storage and processing, in the article various issues have been identified but we had given the research directions only for two issues such as variable block size allocation and file context based replication. The proposed methods definitely

give the better memory allocation strategies and greatly reduce the memory wastage. The current scenario of the memory usage is requiring these kinds of proposals and we feel the methods are significant in the field of big data.

References

- [1] Uma Pavan Kumar Kethavarapu, “Various Computing models in Hadoop eco system along with the perspective of analytics using R and Machine learning”, Vol. 14 CIC 2016 Special Issue International Journal of Computer Science and Information Security (IJCSIS), PP-17-23.
- [2] Uma pavan kumar kethavarapu, “The Ten Ingredients of Data Base Systems for Improving Performance and Their Review Leading to Research Problems”, IFRSAs International Journal Of computing—Vol2—issue 2 April 2012, 409-415.
- [3] S. Madden, “From Databases to Big Data.”, IEEE Internet Comput., vol. 16, no.3, 2012.
- [4] P. Zikopoulos, C. Eaton, and others, Understanding big data: Analytics for enterprise class hadoop and streaming data. McGraw-Hill Osborne Media, 2011.
- [5] A. McAfee, E. Brynjolfsson, T. H. Davenport, D. J. Patil, and D. Barton, “Bigdata”, Manag. Revolut. Harv. Bus Rev, vol.90, no. 10, pp. 6167, 2012.
- [6] R. Appuswamy, C. Gkantsidis, D. Narayanan, O. Hodson, and A. Rowstron, “Scale-up vs Scale-out for Hadoop: Time to rethink?”, in Proceedings of the 4th annual Symposium on Cloud Computing, 2013, p. 20.
- [7] C. P. Chen and C.-Y. Zhang, “Data intensive applications, challenges, techniques and technologies: A survey on Big Data”, Inf. Sci., vol. 275, pp. 3143-47, 2014.
- [8] J. Y. Monteith, J. D. McGregor, and J. E. Ingram, “Hadoop and its Evolving Ecosystem.”, in IWSECO@ ICSOB, 2013, pp. 5768.

- [9] Jorge Bernardino, Experimental Evaluation of a New Distributed Partitioning Technique for Data Warehouses, 1098-8068/010 2001 IEEE.
- [10] Yanzhen Qu, Improving the Security of the Distributed Enterprise Data Warehouse System, 978-1-61284-4577-0085-9/11 2011 IEEE.
- [11] T. P. Latchoumi, Multi Agent Systems in Distributed Data warehousing, Int'l Conf. on Computer & Communication Technology, 2010.
- [12] K.Srikanth, "Data Warehousing Concept Using ETL Process for SCD Type-2", American Journal of Engineering Research (AJER) e-ISSN: 2320-0847 p-ISSN : 2320-0936 Volume-2, Issue-4, pp-86-91, 2013.
- [13] Pavel A. Smirnov, "Domain Ontologies Integration for Virtual Modelling and Simulation Environments", Procedia Computer Science, Elsevier, Volume 29, 2014, Pages 2507-2514.
- [14] Dr.S. Saraswathi and S. Venkataramanasam Design of dynamically updated automatic ontology for mobile phone information retrieval system Int. J. Metadata, Semantics and Ontologies, Vol. 9, No. 3, 2014.
- [15] Isabelle Guyon, An Introduction to Variable and Feature Selection, Journal of Machine Learning Research, 2013 , p.1157-1182.

