Enhancing Performance of Map Reduce Workflow through H2HADOOP: CJBT

Gopichand G, Vishal Lella, Sai Manikanta Avula

ABSTRACT — Distributed computing uses Hadoop framework for handling Big Data in parallel. Hadoop's confinements can be exploited to execute various activities proficiently. These confinements are for the most part an immediate consequence of locality of data in the cluster, scheduling of tasks and various jobs and allocation of resources in Hadoop. Productive resource allocation remain as a challenge in Cloud Computing Map Reduce stages. Henceforth, we propose H2Hadoop, which is an enhanced architecture which decreases calculation cost related with Big Data analysis. The proposed framework additionally helps in solving the issue of resource allocation in local Hadoop. H2Hadoop provides a reliable, accurate and far faster solution for "text data", such as finding DNA sequences and the theme of a DNA succession. Additionally, H2Hadoop gives an effective Data Mining technique for Cloud Computing environment. H2Hadoop design influences on Name Node's ability to assign jobs to the Task Trackers (Data Nodes) inside the group. Building a metadata table containing information of location of data nodes with required features that are needed when in future a similar job is requested to the job tracker then it should compare the metadata and CJBT for assigning data nodes which were previously assigned instead of storing and reading through whole gathering again. Contrasting with local Hadoop, H2Hadoop diminishes time taken by CPU, number of read tasks, and other Hadoop factors.

Key Words: CJBT, DNA sequences, H2Hadoop.

I. INTRODUCTION

Parallel handling in Cloud Computing is developing day by day as an interdisciplinary territory for research as a result of heterogeneous nature and extensive size of data. Making an interpretation of successive data to significant information requires generous computational power and effective calculations to observe the level of resemblances among numerous sequences. Successive pattern mining or data analysis applications, for example, DNA arrangement aligning and theme finding normally require extensive and complex measures of data processing and computational abilities. Productively focusing on and scheduling of computational resources is required to tackle such complex issues. Although, data indexes portions can be read by humans, it is extremely mind boggling to understand and process utilizing conventional preparing strategies. Accessibility of source, which is open and parallel processing stages in distributed computing, has given new path for research and investigate composed of semi-sorted or unstructured data. Data collected currently is so huge and vastly growing day by day which is beyond storage capacity and processing power. The computation process related with this data needs proper fetching, storing and processing. Data from sensors, social media like Facebook, twitter, online shopping, airlines, hospital etc has large variety of data including structured, semi structured and unstructured types, which is growing on a huge scale. Semi structured includes log files and unstructured include all the audio, video, text and images. The 4V's of Big Data are 1) Volume of the data, which varies from giga byte to tera and zeta bytes. 2) Velocity, which tells how fast the fetch store and process computation takes place. 3) Varity of the data that is structured, semi structured and unstructured data. 4) Veracity of the data, it tells about the uncertainty of status of the data or how clear the data is in existence. Different technical challenges in Big Data have been discussed in previous research and additionally, there are numerous difficulties and challenges that have to be resolved. Since Big Data has these issues, it needs such an environment or framework to work through these challenges. This process of computation dealing with fetch, store and process of big data is dealt with Hadoop to beat the above challenges. Different challenges in Big Data have been discussed in previous research, which are mainly technical, for example, the storage of Big Data and diminishing the redundancy. Also there are numerous challenges, for example, cleaning, integrating, representing, aggregating and extracting the data. Since Big Data has these limitations, it needs such a domain or system to work through these difficulties. This process of computation dealing with fetch, store and process of big data is dealt with Hadoop to beat the above challenges. Hadoop Map Reduce jobs applied on DNA dataset has particularly tasks related with the similarity of sequences, super and sub sequences in DNA. Such errands as a rule require different Map Reduce Jobs to get to similar information ordinarily. For a task dealing with matching of a DNA sequence if a n-nucleotide length arrangement present in a particular Data Node, at that point any superstring succession must be found in same Data Nodes. How about we assume that two different clients are hunting down a sequence, which is similar in Big Data source records. When first client finds the required sequence, the second client will similarly experience same advances again to locate similar outcomes. Since the jobs are independent and free of each other clients do nott share the results. Thus, it leads to redundancy of Process, which remains a noteworthy and an issue yet to be solved in local Hadoop Map Reduce framework.

Revised Manuscript Received on April 05, 2019.

Gopichand G, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India. (Email: gopichand.g@vit.ac.in)

Vishal Lella, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India.

Sai Manikanta Avula, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India.

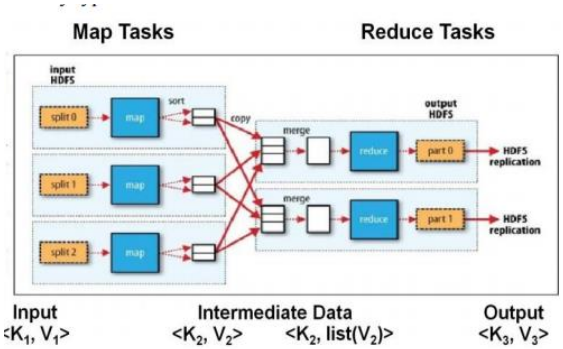

Fig1. Job Execution in Map Reduce

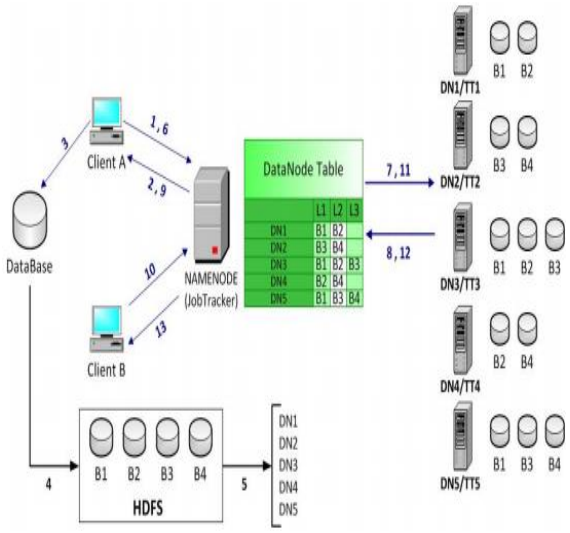

Fig 2: Native Hadoop Map Reduce Workflow

II. RELATED WORKS

In the survey of existing models related to this work, each model had their own techniques that had their own pros and cons. The technique dealing with virtual server technology had flexible, efficient and less expensive information processing solutions. However, these algorithms are not ready to store data that is obtained from the real world due to heterogeneity of the used data. The technique dealing with Map Reduce algorithms and programming models was easy to use even for the naïve users without much involvement with these processing systems and these current centralized designs could effectively deal with this immense volume of data. Nevertheless, enterprises face the challenge of processing these huge sets of data and bandwidth is a scarce resource that should be used properly in the network. The technique involving performing automatic parallelism, circulation over a group of machines, provided a far efficient and cost effective mechanism for data processing using Map Reduce as another programming system. However, its reliability came into question. The technique dealing with MySQL, AQUA that generated a sequence of Map Reduce jobs had a query optimizer that created an effective query plan and optimization depending on the intended model. However, it falls short of intended, desired approach and capability.

Although many of the proposed solutions have been discussed and executed perfectly either by utilizing specific data or by applying them under a few conditions, some of these solutions are not applicable as expected when they are being deployed in a real network such as Hadoop Cloud. When applying such solutions, the runtime analysis and debugging of that process is not easy to be addressed and monitored by using the traditional approaches and techniques. In order to overcome the limitations of Hadoop map reduce regarding the query optimization, redundancy, cpu time, number of read operations we propose enhance Hadoop map reduce architecture which has the very crucial common jobs block table present.

III. PROPOSED SYSTEM

To propose H2Hadoop, which is an upgraded and improved Hadoop design that decreases the calculation cost related with Big Data analysis. The proposed design also addresses the allocation of resources in native Hadoop. Proposed Hadoop Map Reduce workflow (H2Hadoop) is the similar to the native Hadoop as far as equipment, system, and hubs. The software level has been enhanced for increasing Hadoop performance. The name node in enhanced version has CJBT table, which stores the data nodes assigned of the previous jobs which is useful when in future the client with similar related job comes to the job tracker. Then directly job tracker checks and confirms with the CJBT table and assigns the data nodes, which were previously assigned to the similar job. The set of objectives can be categorized into three modules: Data Preprocessing Module: Information preprocessing is a capable apparatus that can empower the client to treat and process complex information; it might expend a lot of handling time. Data Migration Module: After fetching it, the storing part of computation happens in this module in HDFS. Data Analytic Module: The map reduce part of computation happens in this module that involves mapper and reducer and data is dealt with key value pairs.

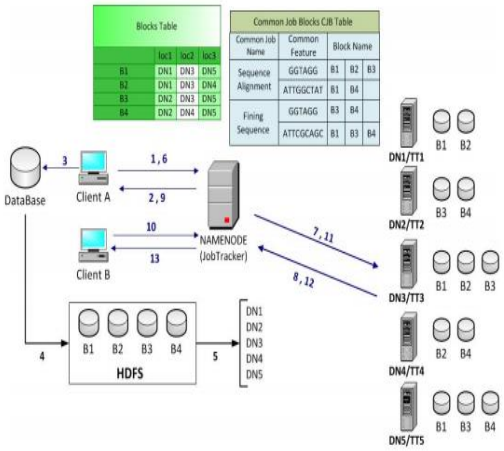

Fig 3: H2Hadoop Map Reduce Workflow

Interpreting the data to significant information requires considerable work on computation as it involves fetching, storing and processing and productive calculations for identifying the level of similarities between multiple sequences. Successive data applications, for example, DNA sequence adjusting for the most part require substantial and

complex measures of data processing and computational abilities. Effectively focusing on and scheduling of computational resources is additionally required to such intricate issues. Some of these data are readable by human, then also it can be exceptionally complex to be comprehended and processed utilizing the conventional strategies. Accessibility of source, which is open and parallel processing stages in distributed computing, has given new path for research and investigate composed of semi-sorted or unstructured data. In native Hadoop, the name node keeps a track of data nodes through heartbeat and block table in HDFS. Name Node is in charge of appointing the jobs to a client and separating that activity into tasks. Name Node additionally assigns the tasks to the Task Trackers (Data Nodes). Knowing which Data Node holds the blocks containing the required data, Name Node should have the ability to manage the occupations to the specific Data Nodes without encountering the whole gathering. In the enhanced Hadoop, a pre-processing phase is implemented in the Name Node before assignment of tasks to data nodes. Building a metadata table containing information of location of data nodes with required features that are needed when in future a similar job is requested to the job tracker then it should compare the metadata and CJBT for assigning data nodes which were previously assigned instead of storing and reading through whole gathering again.

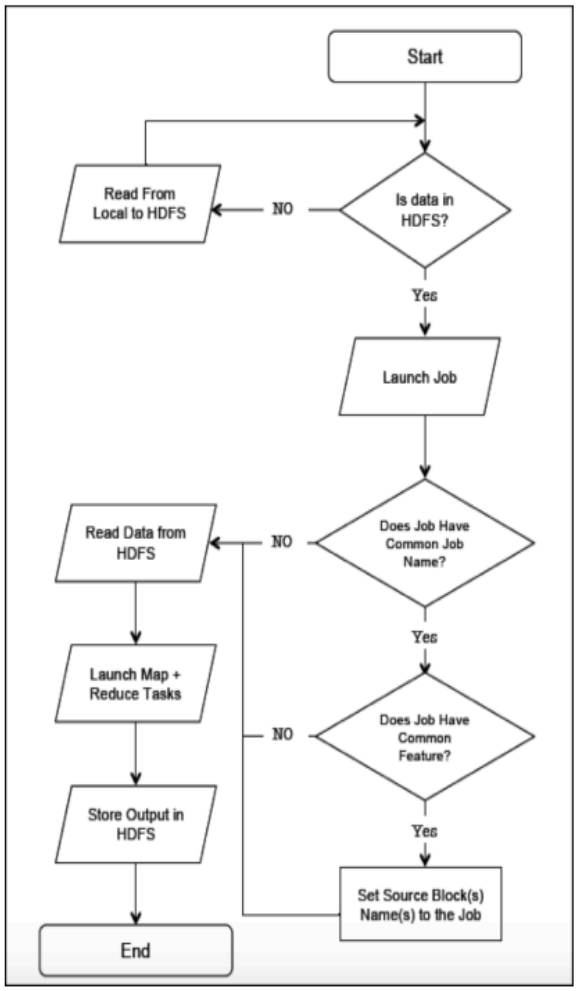

Fig 4: H2Hadoop Map Reduce Workflow

A preprocessing phase happens in Name Node so that job tracker can compare the metadata and CJBT and check if the new incoming job in future is same as the previous job in past. Above given flowchart shows the process of the proposed implemented architecture works. CJBT stores common job name, features and blocks that further when matched with previous jobs then it verifies with block table to get the data nodes which were previously assigned to similar job in past. This way the storing part of job reduces and only processing part occurs which decreases the CPU time. Further, we can decrease the CPU execution time by removing duplicates of the searched sequences in blocks to get extension blocks which helps in decreasing time for searching sequence.

COMMON JOB BLOCKS TABLE COMPONENTS

Common Job Name	Common Feature	Block Name		
Sequence_Alignment	GGGATTTA	B1	B2	B3
	TTTAGA	B1	B4	
Fining_Sequence	TTTAGCC	B3	B6	
	GCCATTAA	B1	B3	B4
	AATCCAGG	B3	B5	

Fig 5: Example notation of CJBT components

IV. EXPERIMENTAL RESULTS

Results were collected from the experiments after the execution of following steps.

1. Collect and upload dataset on DNA sequence. 2. Run Hadoop Map Reduce and search required sequence. 3. Improve Hadoop performance by using metadata of related jobs, instead of searching the whole cluster again by running H2hadoop and then searching the sequence. 4. Analyze performance comparison graph. 5. Adding on my extension work to further increase the performance and further analyze the comparison graph.

We talked about the proposed work process in improved architecture and contrasted the performance with native Hadoop. In H2hadoop, we read less information, so a portion of the Hadoop factors, for example, number of read tasks, CPU time and number of bytes read are reduced. The name node in enhanced version has CJBT table that stores the data nodes assigned of the previous jobs which is useful when in future the client with similar related job comes to the job tracker. Then directly job tracker checks and confirms with the CJBT table and assigns the data nodes that were previously assigned to the similar job.

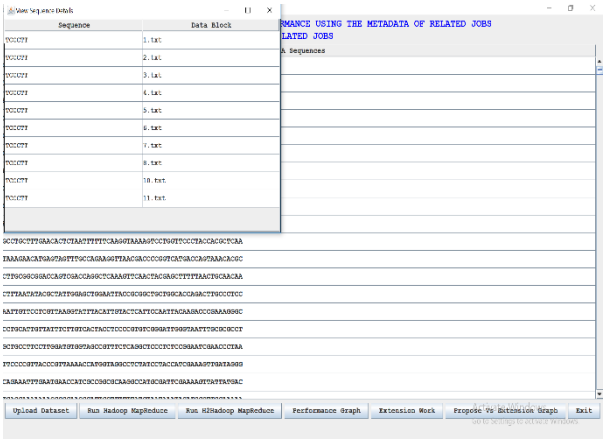

Fig 6: Sequence and its corresponding data blocks

Fig 7: Performance comparision graph between Hadoop and H2Hadoop

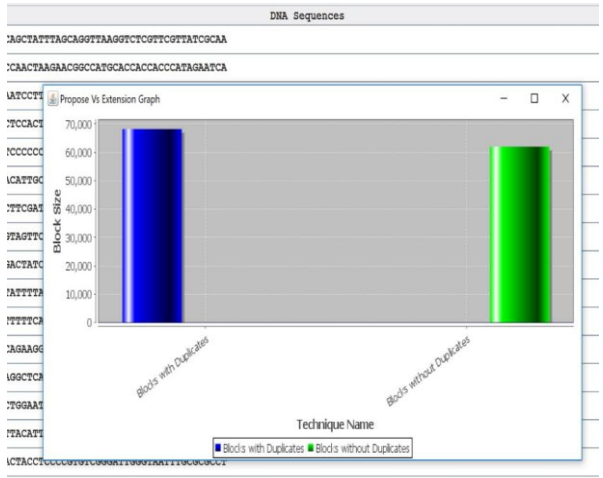

Fig 8: Decrease in block size in our extension work which further increases performance by reducing time for searching a sequence

V. CONCLUSION

In our research work, we compared the expected performance of proposed architecture to that of Native Hadoop and as well decreasing the sequence searching time by removing duplicates. The name node in enhanced version has CJBT table, which stores the data nodes assigned of the previous jobs which is useful if there is a case in future that the client with similar related job comes to the job tracker. Then the job tracker directly checks and confirms with the CJBT table and assigns the data nodes, which were previously assigned to the similar job. The proposed system additionally lessens the data transfer within the system, the cost of execution related with Map Reduce job.

The proposed framework has few restrictions. At present, Enhanced Hadoop architecture more works on text data that contain patterns that clients try to search and find frequently. The proposed Hadoop implementation deals with and works on top of Hadoop and gives adequate results. However, if the enhanced Hadoop is coded as a component of the core of native Hadoop, we may be further able to increase performance of the enhanced Hadoop.

In continuation of this research, there is always a scope for new ideas in future. The following can be investigated in the future:

• Improving the implementation part, Coding Enhanced Hadoop as a part of the core of Native Hadoop.

• Consider other data types, for example, numerical data, tabular data.

• Improve UI to be friendlier than the current interface in Hadoop using command line interface.

REFERENCES

1. M. Meng, J. Gao, and J.-j. Chen, "Blast-Parallel: The parallelizing implementation of sequence alignment algorithms based on Hadoop platform," in sixth *International Conference on Biomedical Engineering and Informatics* (BMEI), pp. 465-470, 2013.
2. M. C. Schatz, B. Langmead, and S. L. Salzberg, "Cloud computing and the DNA data race," *Nature biotechnology*, vol. 28, pp. 691-693, 2010.
3. E. E. Schadt, M. D. Linderman, J. Sorenson, L. Lee, and G. P. Nolan, "Computational solutions to large-scale data management and analysis," *Nature Reviews Genetics*, vol. 11, pp. 647-657, 2010.
4. K. Farrahi and D. Gatica-Perez, "A probabilistic approach to mining mobile phone data sequences," *Personal Ubiquitous Computing*, vol. 18, pp. 223-238, 2014.
5. V. Marx, "Biology: The big challenges of big data," *Nature*, vol. 498, pp. 255- 260, 2013.
6. S. Lohr, "The age of big data," *New York Times*, vol. 11, 2012.
7. M. Chen, S. Mao, and Y. Liu, "Big Data: A Survey," *Mobile Networks and Applications*, vol. 19, pp. 171-209, 2014.
8. H. Jagadish, J. Gehrke, A. Labrinidis, Y. Papakonstantinou, J. M. Patel, R. Ramakrishnan, et al., "Big data and its technical challenges," *Communications of the ACM*, vol. 57, pp. 86-94, 2014.
9. T. White, *Hadoop: The Definitive guide* "O'Reilly Media, Inc.", 2012. 74
10. A. B. Patel, M. Birla, and U. Nair, "Addressing big data problem using Hadoop and Map Reduce," *in International Conference on Engineering* (NUiCONE), Nirma University, pp. 1-5, 2012.
11. M. Hammoud and M. F. Sakr, "Locality-Aware Reduce Task Scheduling for MapReduce," in *IEEE Third International Conference on Cloud Computing Technology and Science* (CloudCom), pp. 570-576, 2011.
12. J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, pp. 107-113, 2008.
13. F. Li, B. C. Ooi, M. Tamer, #214, zsu, and S. Wu, "Distributed data management using MapReduce," *ACM Computing Survey*, vol. 46, pp. 1-42, 2014.
14. W. Xu, W. Luo, and N. Woodward, "*Analysis and optimization of data import with Hadoop*," pp. 1058-1066, 2012.
15. J. B. Buck, N. Watkins, J. LeFevre, K. Ioannidou, C. Maltzahn, N. Polyzotis, et al., "SciHadoop: Array-based query processing in Hadoop," in *International Conference for High Performance Computing, Networking, Storage and Analysis* (SC), pp. 1-11, 2011.

16. T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, K. Elmeleegy, and R. Sears, "MapReduce Online," in NSDI, pp. 20-35, 2010.
17. A. Pavlo, E. Paulson, A. Rasin, D. J. Abadi, D. J. DeWitt, S. Madden, et al., "A comparison of approaches to large-scale data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pp. 165- 178, 2009.
18. A.H.Zookeeper,"http://hadoop.apache.org/zookeeper/," accessed Feb 2015. 75.
19. F. Wang, J. Qiu, J. Yang, B. Dong, X. Li, and Y. Li, "Hadoop high availability through metadata replication," presented at the *Proceedings of the first international workshop on Cloud data management*, pp 37-44, 2009. *and Informatics* (ICCCI), pp. 1-8, 2013.