ISSN 2278-3091
Volume 8, No.5, September - October 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse33852019.pdf
https://doi.org/10.30534/ijatcse/2019/33852019

Optimizing Joins in a Map-Reduce for Data Storage and Retrieval Performance Analysis of Query Processing in HDFS for Big Data

Dr.S.Sudhakar[1], Dr.N.Satheesh[2], Dr.S.Balu[3], Amireddy Srinish Reddy[4], Dr.G.Murugan[5]
[1]Professor, Dept. of CSE, Sri Shakthi Institute of Engg. and Tech., Coimbatore – 641062, Tamil Nadu, India
sudhasengan@gmail.com
[2]Professor, Dept. of CSE, St.Martin's Engineering College, Hyderabad, India
nsatheesh1983@gmail.com
[3]Associate Professor, K.S.Rangasamy College of Technology, Tiruchengode-637215, TamilNadu, India
balus@ksrct.ac.in
[4] Assistant Professor, Dept. of CSE, St.Martin's Engineering College, Hyderabad, India,nsatheesh1983@gmail.com
[5] Professor, Dept. of CSE, Vidyalankar Institute of Technology, Wadala East, Mumbai- 400037, India
gopalmurugan0@gmail.com

ABSTRACT

Big Data challenges prompting by the fast sending of investigation of information internationally. New excellent presentation systems are presently required to process a regularly expanding volume of information from informational collections. With information immovably close by and with the capacity given by Big Data Technologies to viably store and examine this information, we can discover answers to these inquiries and work to enhance each part of our conduct. The Hadoop structure is an open-source execution of the Map Reduce (MR) figuring model that is picking up force for Big Data investigation in smart matrix applications. For the most part, different DB2 languages turn into a critical application territory for Hadoop by utilizing Query handling. The outcomes demonstrate a hard belief on the measure of reducers and IO execution of the group, which describes the regular assessment that MR is IO-bound. These outcomes can look at the execution conduct of various languages and fill in as a reason for understanding the impact of design parameters on the last execution.

Key words: Hadoop, Map Reduce, Query Processing, Performance, Data Analysis, Big Data.

1. INTRODUCTION

The most mainstream open-source accomplishment of a single computing hub (Apache Hadoop, Wiki) [1] is the Hadoop. Hadoop and MR projects are utilized in managing a gigantic measure of information. Hadoop can be used for accumulating vast knowledge and for handling information, for example, information mining, report examination, record investigation, web ordering, and bioinformatics explore. As the name suggests, *"Huge Data"* exhibits a few difficulties to Information System experts.

For proficient exchange processing, most DBMSs are intended for including, refreshing, looking for, and recovering limited quantities of data in an enormous database. It gives the idea that stages have been made to manage the mass and structure of Big Data. A Hadoop framework comprises of two noteworthy segments. The principal portion is the Hadoop MR engine – MR. The subsequent layer is HDFS – Hadoop Distributed File System, which is motivated by Google's GFS (Google File System) [2].

HDFS partitions record into segments that are simulated among a few distinctive registering hubs with no thoughtfulness regarding whether the layers are isolated equally. At the point when a task is started, the processor of every hub works with the information on their nearby hard disks. In the end, when the massive document is gotten to, high accumulated I/O data transfer capacity can be accomplished by getting to the various hubs in parallel.

The MR [3] programming structure can improve the complicated nature by running parallel information preparing capacities over numerous computing hubs in a bunch as versatile MR causes software engineers to disperse programs crosswise over centers as they as executed in parallel. MR consequently accumulates results from the numerous hubs and returns a solitary outcome or set. Significantly, MR stages offer adaptation to internal failure that is altogether straightforward to software engineers. The accompanying figure 1 is spoken to the MR information investigation.

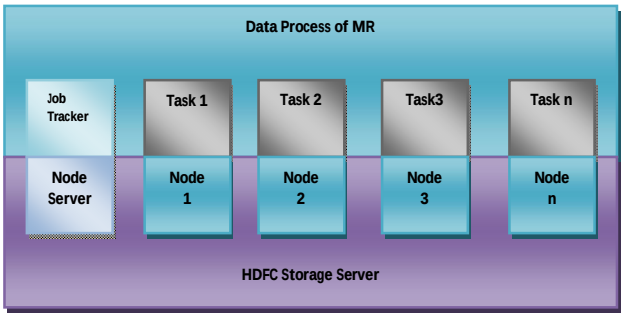

Figure 1: The process model of HDFC and MR data analysis

The exhibition of the group can be improved by Hadoop because various hubs work simultaneously to give high throughput. This makes MR a convenient and appealing programming model for parallel information preparing in superior group processing atmospheres. This paper utilizes a Hadoop-based information investigation biological system to help a Big Data application with vital information to test the presence of gigantic details after sending in both traditional datasets [4].

2. RELATED WORKS

In the 1990s, Google needed to think of more information and to get the best possible arrangement, and it has taken 13 years. In 2003, they had presented GFS (Google File System), which is a method [5] to store immense information. In 2004, they gave MR, which is the best handling method. They have distributed a "white paper," which has a depiction of GFS, Yahoo, which is the following best search engine after Google presented HDFS in the year 2006-2007, and MR was introduced in 2007-2008. They have taken the white paper, which was given by Google, and began executing and thought of HDFS (Hadoop Distributed File System) and MR. These are the two center segments of Hadoop. Hadoop was then presented by Doug Cutting in 2005 [6].

The informational collections for Hadoop MR are put away in the HDFS in information squares, which can be handled promptly by Map errands incomparable. The whole Map Reduce system works on (key, esteem) sets [7], and discrete maps assignments never speak with one another. Each guide was undertaking procedure information logs in remoteness. It is regularly the situation that a document is a distinct fresh in a delimited bit of information, even though this viewpoint is exceptionally adaptable. The guide task at that point puts zero to many average yield records. Alternatively, the yield records structure the guide assignments might be sent through a combiner that merges the files nearby to a guide task into a littler arrangement of identical logs before being referred to the MR [8]. This association is actualized utilizing a similar capacity as the decrease task. The documents are arranged by key, and all qualities with a comparable key are prepared together by the same reducer in no specific request. The MR stage can't begin until the guide stage has finished, and every MR produces zero to many yield records for the MR work. The information types utilized as the contributions to some random stage shouldn't be equivalent to the yields.

Moreover, the application isn't required to utilize the information keys for anything. It is additionally the situation that an MR occupation may not need both an MR stage. Numerous employments use a personality guide errand and necessarily process information in the diminish assignments or the other way around [9].

Query languages based on various basis and information structure ideas. In the Hadoop setting, the most usually applied approach is called DB2 putting together its name concerning the distinctions to exemplary social database frameworks that use SQL [10]. The two most normal instances of such inquiry dialects are Pig2 and Hive3. In this paper, we use another developing language called Cascalog4 because of its solid match for the test inquiries.

3. PROPOSED METHODOLOGY FOR MR

This segment gives the plan, and engineering of Hadoop, the comprehensive data of what devices have been utilized, and the all outnumber of information gathered from various assets. The info used in this paper is in an organized arrangement and contains a large number of records, which are as HIVE table and in MySQL tables. Hadoop keeps running with the assistance of various apparatuses and systems, which are likewise talked about in point by point. To play out the examination, the equipment and programming necessities are required; the measure of room and memory needed for every one of the Virtual Machines is gathered [11].

3.1. Proposed Design of the Study

The accompanying outline clarifies the design of Hadoop utilized for this work. It is focused on the HDFS document framework, which is used to store the information. To accomplish the ideal parallelism, MR, and structure are utilized to process HDFS information and give an asset to the board. Apache HIVE is based on Hadoop to provide an information rundown and investigation on HDFS information. Apache SQOOP [12] is utilized to move data between relational databases and the HDFS framework.

At last, when the data is stored in both DB2 and HIVE databases, an examination is performed on the information, and the outcomes are thought about. A drawing that portrays the procedure that was pursued to embrace the trial examination shows up underneath the proposed design figure 2. Both theDB2 database and the HDFS were kept running on comparable equipment inside a similar cloud. Be that as it may, the HDFS framework was circulated over a few hubs. As would be reasonable, the HDFS framework performed better in all the test preliminaries. The fundamental thought of the task is to analyze the presentation of DB2 [13] and HIVE and to demonstrate that with enormous informational indexes, HIVE gives better execution that the conventional database. The design chart clarifies the procedure, which will be followed in this task. The colossal information gathered Twitter is changed into DB2 in an organized configuration. Utilizing SQOOP, which

is a piece of information relocate device, the data from DB2 is stacked into the Hadoop appropriated record framework. The data is then moved to HIVE as organized tables. Utilizing HIVE Query Language (HQL) [14] [15], a few inquiries are performed on the information and the opposite side; using DB2, similar questions are performed.

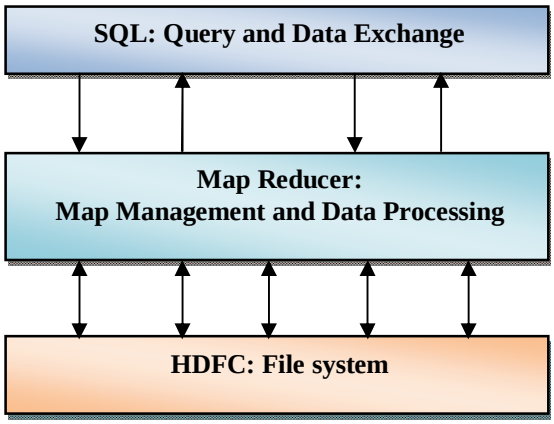

Figure 2: Proposed Architecture Diagram

This proposed system is utilized for task scheduling and cluster source organization. MR structure is used to part information into little pieces and executes the related employments on hubs [16,17]. The outcomes will be gathered from centers, incorporated, and after that arrival to clients. Along these lines, MR changes a solitary hub preparing occupation to parallel handling employment to improve the execution productivity. The flowchart of the proposed is indicated in figure 3.

Input	:	Data Set →D_1, a
Data Map	:	i_1, a_1→i_2, a_2
Data Mix	:	i_2, Iteration a2 → i_2, a_2
Divider	:	i_2, a_2 → int
Reducer	:	i_2, Iteration a_3→i_3, a_3
Output	:	i_3, a_3 → Data Set

3.2. Map Reduce Function

$$MR_x(X) = \{x \in U : \mu_X^{MR}(X) = 1)$$

$$MR_x(X) = \{x \in U : \mu_X^{MR}(X) = 1)$$

$$MRN_x(X) = \{x \in U : \mu_X^{MR}(X) > 1)$$

3.3. Membership Function

$$\mu_X^{MR} U \to \{0,1\}$$

$$\mu_X^{MR}(x) = \frac{X \cap MR(x)}{MR(x)}$$

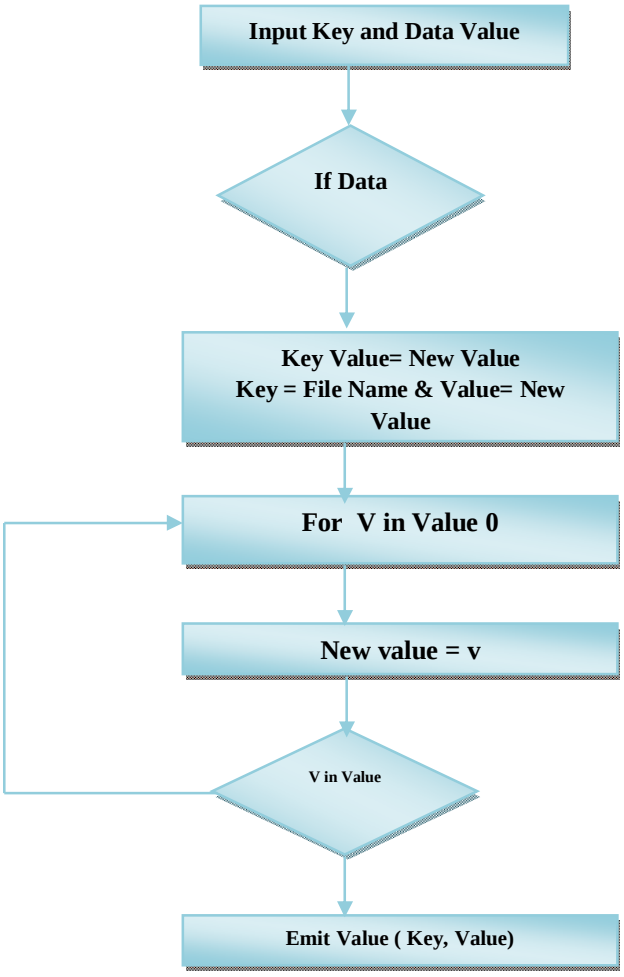

Figure 3: Flowchart for Proposed MR

3.3. Test data set for data analysis

The examinations were achieved for 5 dissimilar sizes of datasets. It compared to wellsprings of information created for 100, 200, 300, 400, and 500. The data was changed to acquire the structure practically equivalent to Abadi that was the most advantageous from the perspective of inquiry performance. The littlest record size was 1 GB for one of the documents in Query 2 for 500 sources. The most significant record was 10 GB in Query 14 for 500 sources. In the heap test, log just on the heap times of records more prominent than 1GB to 50 GB to increase the clearness of the chart [18, 19].

3.4. Query processing execution time

The test questions: 1-15 were executed and presented in table 1. We chose to use this question verbal as it appeared to have regular communication with the first request. Be that as it may, those inquiries could be additionally effectively executed utilizing Hive. Some exhibition contrasts could be standard, and they are mostly secured. For every one of the situations, information was first stacked from HDFS and further read and composed from HDFS.

Table 1: Performance of average time for data analysis

Test Data Size *(in GigaByte)*	No. of Data Node	Query Processed Time *(in ms)*
1 GB	2	4.58
2 GB	4	3.78
3 GB	6	2.98
4 GB	8	2.89
5 GB	10	2.15

4. RESULT AND DISCUSSION

In this segment, current and depict the presentation of an inquiry with great choosiness in figure 4. Basing on every one of the data in the paper, one can see that most considerable presentation enhancement can be seen up to10 hubs, after what they become moderately little. 10 GB of information was produced by utilizing the order. The direction took 18 minutes and 20 seconds to execute.

$Hadoop \rightarrow jar \rightarrow Hadoop \rightarrow Test.jar \rightarrow Teragen / TeraSort / Input$
$Hadoop \rightarrow jar \rightarrow Hadoop \rightarrow Test.jar \rightarrow Teragen / TeraSort / Output$

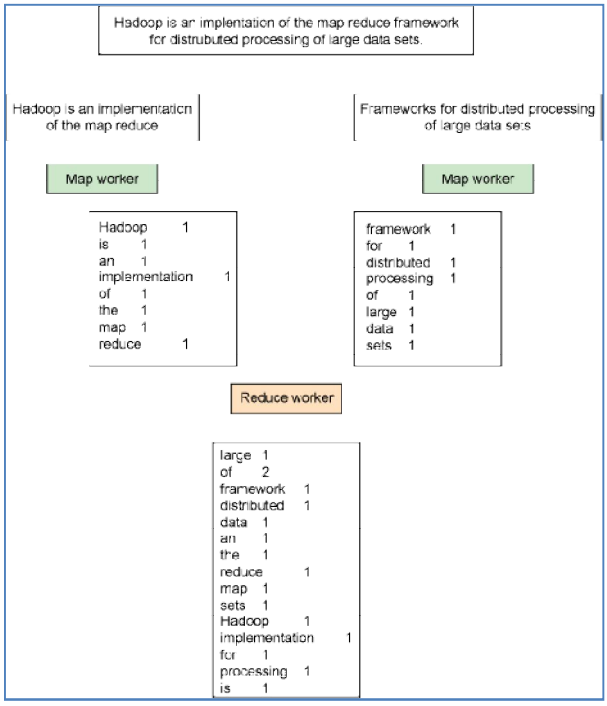

Figure 4: Implementation of Map Reduce

Table 2. Computation of Execution Time

Data Size (in GB)	Standard HDFC (in sec)	Proposed HDFC (in sec)
1	2090	1005
2	3010	2100
3	4050	2899
4	5010	2500
5	6050	4500

There is a noteworthy decrease in the finishing time taken while utilizing the proposed situation approach as it showed the outcomes (Refer to Table 2). The estimated rate decrease, when contrasted with normal HDFS, is 18.28%, and HDFS stability is 6.78%.

4.1. Web Server Loading Performance

A load time overhead examination was presented in this section. The enduring capacity default administration for AWS and its presentation can affect the execution of the entire application, particularly significant for AWS based applications. Figure 5 Presents consequences of stacking documents with sizes going from 1 to 10 GB on an m1.small examples relying upon the measure of hubs. The highest relative exhibition increase is seen in the range up to 8 hubs; however, further hubs likewise improve execution.

4.2. Data Query Process Execution

This analysis attempts to examine and analyze inquiry execution times among RDBMS and Hadoop. Analysis is finished with expanding information sizes (See Table 3) to investigate if and where Hadoop gains effectiveness over RDBMS with its disseminated and parallel preparing engineering. Execution is assessed for two queries.

- ***Select*** count (picture ID) ***from the*** evaluations ***group by*** user name.
- ***Select*** count (picture ID) ***from*** evaluations ***where*** appraisals like '20.'

This experimental evaluation embraces the way that Hadoop runs information examination in parallel over the hubs in the group. Hadoop can circulate mix, sort movement on various hubs accessible in the group demonstrates to pick up execution over running information examination query in conventional RDBMS set up given in Figures 5 and 6.

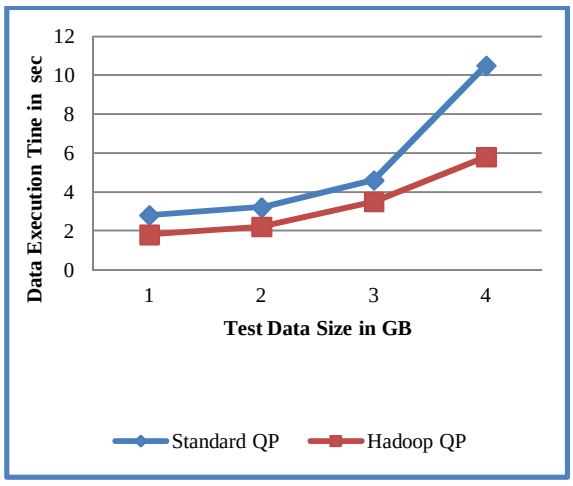

Figure 5: Performance of Query Processing Execution Time (by user)

Table 3: Computation of Execution Time for data analysis in Query Processing

Data Size (in GB)	Standard QP (in a sec)	Proposed Hadoop QP (in a sec)
1	1.99	2.25
2	4.25	3.12
3	8.19	3.99
4	10.65	5.56
5	12.25	7.25

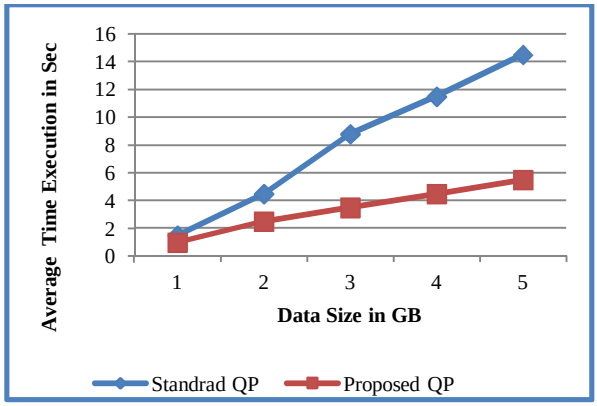

Figure 6: Performance of Query Processing Execution Time (by rating)

4.3. High Selectivity Performance

To depict the exhibition of a query with high selectivity is obtainable in this segment. Basing on every one of the figures in the paper, one can see those most excellent presentation enhancements can be seen up to 10 hubs, after what they become generally little. By contrasting the exhibition of high selectivity, the situation isn't fundamentally reliant on the measure of reducers. Simultaneously using hubs with high IO execution furnishes huge advantages with time diminishing of the degree of 2 up to 4 contingents upon the setup in the accompanying figure 7.

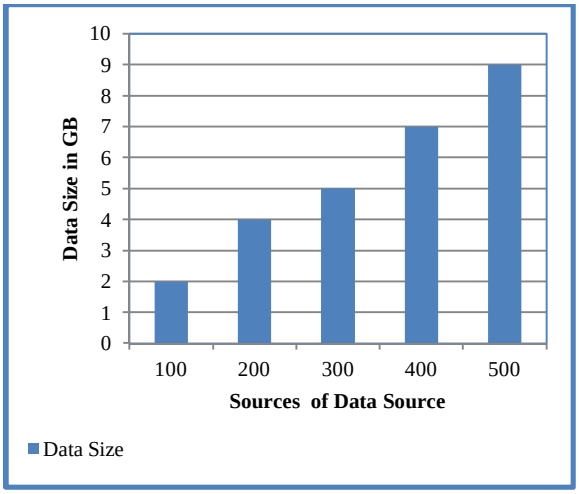

Figure 7: Query executing time from different sources

4.4. Performance of Complex Interdependency

The discussion is about execution query results with sophisticated interdependency design between documents. As a matter of first importance, well-built reliability on the measure of reducers can be watched where the exhibition diminishes for more than 5 hubs. The purpose behind it may be the more critical requirement for correspondence among mappers and the reducer because of the multifaceted nature of the inquiry, regardless of generally littler data sources. In any case, insufficient transfer speed is accessible because of low IO execution. With more reducers, implementation is radically expanded.

Furthermore, additional reliance on the IO execution of the group can be watched. Aside from the common lessening in times among little and enormous illustrations as saw in the anterior segment, other impacts can be taken note. We can see that high IO execution permits to somewhat making up for the reality of having only one MR by successful correspondence with it. Further, the expanded measure of MR in such a case grows execution; be that as it may, to a generally lesser degree as on account of low IO execution.

4.4.1. Low Selectivity Performance

Here current and examine execution for question with small discrimination, so adequately exceptionally high result obtained. Most importantly, comparable patterns in the past areas can be taken note. In any case, after closer assessment, we watch the distinction between IO execution concerning system execution and circle execution. Note that the accompanying examination puts together just concerning the introduced information. Further tests with exact plate and system observing ought to be performed to affirm the ends.

While looking at one-reducer designs, and notice that arrangement with m1.large occurrences with high IO execution can maintain a strategic distance from execution decline with the high developing number of hubs. In this circumstance, more top system execution becomes effective. Be that as it may, it doesn't demonstrate any advantages of those extra hubs. This can be brought about by the impediment of the disk execution, as just a single reducer needs to compose a moderately enormous yield document to the HDFS. In such a case having a more massive amount of less expensive machines with lower IO execution can give somewhat better outcomes.

5. CONCLUSION AND FUTURE WORK

For query processing, we exhibited a thorough execution examination of the Hadoop arrangement in this paper. It demonstrated that particularly on account of complex inquiries, the measure of reducers and the IO organize the execution of the group are significant. In the event of investigations with exceptionally high yield, IO disk execution is the constraining element; however, it very well may be improved by applying more reducers.

Hadoop task execution utilizing default parameter settings has prompted execution issues. We have exhibited HDFC to improve Hadoop implementation in this paper by consequently tuning its arrangement parameters. The

streamlined Hadoop system can be used for an adaptable examination of huge informational collections. The HDFC accomplished a most extreme computational speedup of 28.12 *sec*. Times quicker than the successive information and 1.68 *sec*. Times faster than a parallel information investigation. At present, the Hadoop structure is profoundly appropriate to disconnect adaptable information investigation. In any case, the high preparing overhead connected with info and yield documents restricts the use of Hadoop to an on-line survey of information streams. Further investigation will apply in-memory preparing strategies to empower constant information stream examination for power framework applications.

References

1. He, B., Fang, W., Luo, Q., Govindaraju, N.K., & Wang, T.,**Mars: a Map Reduce framework on graphics processors**. In Proceedings of the 17th international conference on Parallel architectures and compilation techniques (pp. 260-269). ACM, 2008 https://doi.org/10.1145/1454115.1454152
2. Yang, G., **The application of Map Reduce in the cloud computing, In Intelligence Information Processing and Trusted Computing (IPTC**).2nd International Symposium on. Hubei, RPC, 22-23 Oct. IEEE,2011. https://doi.org/10.1109/IPTC.2011.46
3. Quintero, D., Navarro, E. A., Garro, P.B., Castro, R.C. F.D., Huertas, L.C.C., Jiang, P., &, J, **Implementing an IBM InfoSphere Big Insights Cluster Using Linux or Power,** 2015.
4. Weber, S., **Big Data Privacy and Security Challenges.** Proceedings of ACM Workshop on Building analysis datasets and gathering experience returns for security, ACM pp. 1-2, 2012. https://doi.org/10.1145/2382416.2382418
5. Liu, Huan, and Dan Orban. Gridbatch: **Cloud computing for large-scale data intensive-batch applications**, In Cluster Computing and the Grid, CCGRID'08. 8th IEEE International Symposium on, pp. 295-305. IEEE, 2008. https://doi.org/10.1109/CCGRID.2008.30
6. Afrati, Foto N., and Jeffrey D. Ullman, **Optimizing joins in a map-reduce environment,** In Proceedings of the 13th International Conference on Extending Database Technology, pp. 99-110. ACM, 2010.
7. Shvachko, Konstantin, Hairong Kuang, Sanjay Radia, and Robert Chansler, **The Hadoop distributed file system,** In Mass Storage Systems and Technologies (MSST),IEEE 26th Symposium on, pp.1-10. IEEE, 2010. https://doi.org/10.1109/MSST.2010.5496972
8. P. M. Ashton, G. A. Taylor, M. R. Irving, I. Pisica, A. M. Carter, and M. E. Bradley, **Novel application of detrended fluctuation analysis for state estimation using synchrophasor measurements**, IEEE Transactions on Power Systems, vol.28, no.2, pp.1930–1938, 2013. https://doi.org/10.1109/TPWRS.2013.2248027
9. Z. Yu, Z. Bei, H. Zhang, **RFHOC: A Random-Forest Approach to Auto-Tuning Hadoop's Configuration** IEEE Xplore: IEEE Transactions on Parallel and Distributed vol. PP, no. 99, p. 1, 2015.
10. P.Trachian, **Machine learning and windowed sub-second event detection on PMU data via Hadoop and the open PDC**, in Proceedings of the IEEE PES General Meeting, PES 2010, USA, July 2010.
11. M. Zaharia, T. Das, H. Li, T. Hunter, S. Shenker, and I. Stoica, **Discretized Streams: Fault-tolerant streaming computation at scale**, in Proceedings of the 24th ACM Symposium on Operating Systems Principles, SOSP, pp. 423–438, USA, November 2013. https://doi.org/10.1145/2517349.2522737
12. M.F. Husain, L. Khan, M. Kantarcioglu, and B. Thuraisingham, **Data-Intensive Query Processing for Large RDF Graphs Using Cloud Computing Tools**, IEEE 3rd International Conference on Cloud Computing, Miami, FL, USA: 2010, pp. 1-10,2010. https://doi.org/10.1109/CLOUD.2010.36
13. M. Farhan Husain, P. Doshi, L. Khan**,** and B. Thuraisingham, **Storage and Retrieval of Large RDF Graph Using Hadoop and Map Reduce**, Cloud Computing, M. Jaatun, G. Zhao, and C. Rong, eds., Springer Berlin / Heidelberg, pp. 680-686, 2009
14. J. Dawei, C.O. Beng, S. Lei, and W. Sai, **The Performance of MapReduce: An In-depth Study**, vol. 3, pp. 472- 483, 2010.
15. J. Myung, J. Yeon, and S. Lee, **SPARQL basic graph pattern processing with iterative MapReduce**, Proceedings of the 2010 Workshop on Massive Data Analytics on the Cloud - MDAC '10, Raleigh, North Carolina, pp. 1-6., 2010. https://doi.org/10.1145/1779599.1779605
16. Sasi Bhan1, Dr. JKR Sastry, P. Venkata Sunil Kumar, B. Venkata Sai, K.V.Sowmya, **Enhancing Performance of IoT Networks through High-Performance Computing**, International Journal of Advanced Trends in Computer Science and Engineering, Vol.8, No.3, pp:432-442,2019. https://doi.org/10.30534/ijatcse/2019/17832019
17. Jessica Velasco1, Jeshurun Rojas2, John Peter Ramos3, Hannah Mhel Muaña4, Keith Louise Salazar, **Health Evaluation Device Using Tongue Analysis Based on Sequential Image Analysis**, International Journal of Advanced Trends in Computer Science and Engineering, Vol.8, No.3, pp:451-457,2019. https://doi.org/10.30534/ijatcse/2019/19832019
18. Sudhakar, S., Chenthur Pandian, S. **Hybrid cluster-based geographical routing protocol to mitigate malicious nodes in mobile ad hoc network**, International Journal of Ad Hoc and Ubiquitous Computing , Vol.21, No.4, pp.224,2016. https://doi.org/10.1504/IJAHUC.2016.076358
19. S.Sudhakar, S.Chenthur Pandian, **Investigation of attribute aided data aggregation over dynamic routing in wireless sensor networks**, Journal of Engineering Science and Technology, Vol.10, No.11, pp.1465, 2015, 2014.