International Journal Of

Recent Scientific Research

ISSN: 0976-3031

Volume: 7(5) May -2016

MITIGATION OF DATA SKEW USING BLOCK CHAIN ALGORITHM

RajaSaranyaKumari R., Lenny Arul Jegan and Yuvasree J

THE OFFICIAL PUBLICATION OF
INTERNATIONAL JOURNAL OF RECENT SCIENTIFIC RESEARCH (IJRSR)
http://www.recentscientific.com/ recentscientific@gmail.com

ISSN: 0976-3031

Available Online at http://www.recentscientific.com

International Journal of Recent Scientific Research
Vol. 7, Issue, 5, pp. 11306-11311, May, 2016

International Journal of Recent Scientific Research

Research Article

MITIGATION OF DATA SKEW USING BLOCK CHAIN ALGORITHM

RajaSaranyaKumari R[1]., Lenny Arul Jegan[2] and Yuvasree J[3]

[1,2,3]Department of Computer Science and Engineering Vel Tech High Tech Dr.RangarajanDr.Sakunthala Engineering College

ARTICLE INFO

Article History:

Received 17th February, 2016
Received in revised form 21st March, 2016
Accepted 06th April, 2016
Published online 28th May, 2016

Keywords:

Map Reduce, Data skew, Hadoop, Block Chain, Web Crawling, Live data Migration.

ABSTRACT

In Big Data, logs of Peta and Tera bytes of data clusters are need to be processed. Hadoop allows these large clusters to be processed by using MapReduce technique, which is a programming tool for processing of data. As a result of MapReduce, Dataskew problem arises because of the static partitioning method followed in traditional Hadoop clusters. This leads to a delay in the overall throughput. To overcome this problem, we propose an innovative concept of Block Chain algorithm, an efficient dynamic data splitting strategy on Hadoop, which monitors the samples while running batch jobs and allocate resources to slaves depending on the complexity of data and the time taken for processing. We also implement Web Crawling to reduce the same, using Hadoop thus eliminating DDOS attack detection scenarios that will happen on the servers we are crawling, which can be done using the distributed systems. This causes the overall output to be enhanced. We implement this project in Hadoop, compare the results with MapReduce technique and our experiment to show that it has negligible overhead and can speed up the execution.

INTRODUCTION

There is a tremendous need for data processing in recent days. Every day, huge logs of Peta and Tera bytes of data are being generated by internet organizations and enterprises [1]. This large data clusters are difficult to be scheduled and processed by a single commodity system. In order to overcome this drawback, Big Data Analytics resulted in Hadoop concept, where the data which is to be processed is distributed among several client/slave machines in order to process. This is carried out by implementing map reduce technique. Map reduce is an effective tool to process large sets of data [1][2]. It is a static way of dividing the job and allocating to client machines for effective processing. There are several popular applications that include MapReduce implementation, such as apache Hadoop, Google map reduce [3][4] etc. The time complexity of the overall process is ultimately dependent on the slowest running task in the case of deadlocks or stragglers [5]. There are several drawbacks of MapReduce in which most prevalent is the Data skew. Dataskew is nothing but the imbalance in the amount of data assigned to each task [6]. The fundamental cause for the occurrence of dataskew is that the user doesn't know the type of Data to be distributed beforehand [7]. This is the speculative problem in MapReduce, which is being solved in the proposed project by the implementation of block chain algorithm, which is a dynamic splitting concept that replaces the MapReduce technique. It comparatively reduces the dataskew problem, which eventually reduces the time complexity thus providing a successful overall output.

Scope

This project will be more effective in terms of time complexities when compared with the traditional MapReduce technique. Also it shows the effectiveness of Hadoop and it will reduce the data skew problem that is more prevalent in the MapReduce. Hence this project using Block Chain will be an efficient replacement for MapReduce. As far as MapReduce is concerned, entire task is ultimately depended on the slowest running job [8]. This straggler process thus eventually increases the time complexity of the final job completion. The delay is significantly caused by the dataskew. The dataskew is the imbalance in the size of the data that is being assigned to the commodity system for processing. Also, the dataskew may occur when a user is unaware of the data type that has to be processed. Dataskew is not prevalent in MapReduce, but is found in parallel database, but on certain joins and operations [9] [10]. Several solutions determined to reduce this problem are more specific to certain applications. When it comes to heterogeneous environment, the dataskew problem goes beyond control, thereby generating poor results [11]. In order to tackle this situation, we propose an innovative technique

**Corresponding author: RajaSaranyaKumari R*
Department of Computer Science and Engineering Vel Tech High Tech Dr.RangarajanDr.Sakunthala Engineering College

based on dynamic data splitting strategy. This technique, named, Block Chain Algorithm reduces the dataskew for reduceside applications.

Compared to earlier works, our contributions include the following

- We propose Web crawling mechanism to show the effectiveness of Hadoop cluster, where web crawling eliminates the DDOS attacks that occur on servers we crawl.
- Query processing done through MapReduce is replaced by an innovative technique called Block Chain Algorithm.
- We implement MapReduce and
- Block Chain to evaluate the performances. Experimental results demonstrate that Block Chain is comparatively beneficial.
- The rest of this proposal comprises of background work on MapReduce, Dataskew, Web Crawling and sampling methods used for the dynamic split of the data, performance evaluation and future work and conclusion.

Background

Map reduce and dataskew problem

In Hadoop, data processing is done through MapReduce technique, wherein the data is split in equal amounts and allocated to the available slaves for processing[1][11][12]. Here, a sampling technique involves a pre-run jobs to be tested for determining the characteristics of the commodity systems [13]. This leads to the delay in the actual real jobs to start processing since traditionally, Hadoop follows this kind of partitioning. The input is transformed into intermediate tuples based on some user defined partitions. Hadoop uses hash partitioner by default. As a result, each reduce tasks copies its partitions and the results are stored locally after processing [14] [15]. In order to preserve the efficiency, every process must finish at same time. But not all the tasks finish at once, and some tasks may unusually take longer to complete thus increasing the overall time complexity of the entire task.

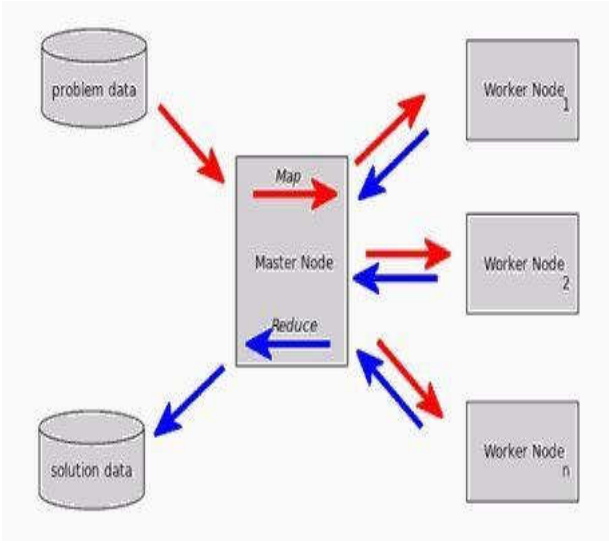

Fig 1 Map Reduce Algorithm

This is because the slaves have different capacities in processing the data. Hence this causes the dataskew problem [14] [16]. In simple terms, dataskew is nothing but the imbalance in the amount of task or data assigned to each node in the cluster. The dataskew can occur in both reduce and map phases. Map skew may occur in the input data but it is easy to be addressed and can be solved by normal splitting methods, whereas, reduce skew is much more difficult since it is arbitrary.

Existing System

The existing LIBRA system with MapReduce, is a Lightweight Implementation of Balanced Range Assignment approach [1] which is used to solve the dataskew for general application. Map Reduce is an effective tool for parallel data processing. As a result of this map reduce data arises most predominantly in distributed systems. In order to solve this problem, LIBRA was adopted. The design goals of LIBRA comprises of Parallelism, Transparency, Total order, Heterogeneity consideration etc. The dataskew is mitigated by adopting several steps. A portion of the real job is sampled to collect the statistics about the intermediate data and the output transmit is known at the end of the normal map process. Then it is the work of the master to accumulate all the results about the data and its types, and decides for partition accordingly. Once the partitioning decision is finalized, the reduce tasks can also be generated without waiting for the entire map process to complete. Although several solutions can mitigate data skew to some extent, they have significant overhead due to the pre-run jobs and are applicable only to certain applications.

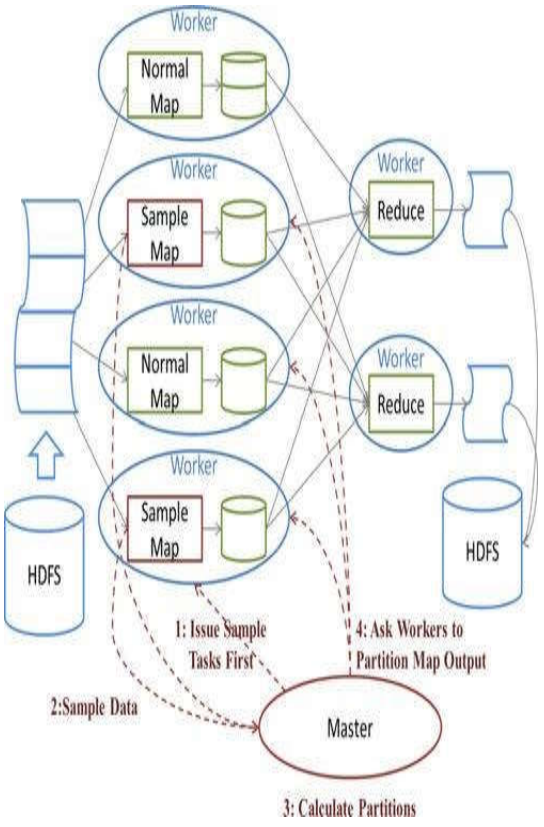

Fig 2 Existing system Architecture

System analysis

Process of sampling and Partition used

In case of unknown information about the datatypes and characteristics of the data and slave nodes respectively, then it will be challenging for one to mitigate dataskew. Two approaches to solve this problem are, to launch a pre-run job to know about the data type and the other is to integrate the sampling along with normal map process. But there are some unresolved issues that resulted in unfavourable outcomes. The sample tasks are being prioritized for execution than the real jobs. This might take a considerable amount of time for execution which will obviously impact the total time taken since, the real jobs go for process only after all the sample tasks finish. Once when the samples and the partition method are ready, the master will notify the slave about the samples and start to process the intermediate data and a separate list of partition is being maintained. After the mapping process, the appropriate partition key are filled up on the list of partition. After the completion of all map tasks, reduce work starts and the list of overall results with the value for all the keys are processed locally and sent to the master.

Heterogeneous Feature Slave Nodes

It is obvious that not all the slaves are equally created and even when same amount of data is allocated, their capacity of processing the data will be different, which is ultimately create impact on the performance throughput.

Problem Definiton And Enhancement

There are certain problems that are discovered from the existing system of LIBRA. Those are

1. Static Data Splitting among nodes.
2. Uneven Capacity of slave nodes.
3. Long Running Jobs.
4. Pre-sampling.

In order to overcome the problems specified we implement an innovative method called Block Chain Algorithm which is based on dynamic data splitting strategy [18] and also adopt Web Crawling to show the improved effectiveness of Hadoop clusters.

Proposed System

We propose an efficient dynamic data splitting strategy on Hadoop which monitors the samples while running batch jobs and allocate resources to slaves depending on the complexity of data and the time taken for processing. We also show the effectiveness of Web crawling using Hadoop eliminating DDOS attack detection scenarios that will happen on the servers we are crawling. Query processing done through MapReduce in traditional Hadoop clusters is an effective tool to process large sets of data. It is a static way of dividing the job and allocating to client machines for effective processing which is replaced by another technique we developed i.e. Block chain query processing and we compare the response times to show its effectiveness. Block chain proved to be as best as MapReduce and can be used in data intense results. Unlike previous work, it does not require any pre-run sampling of the input data or prevent the overlap between the map and the reduce stages wherein the earlier work eventually leads to dataskew since it requires pre-sampling.

Architecture Description

The Tab Separated Values (TSV), Comma Separated Values (CSV) and other resources are provided to the admin. The admin gets the batch jobs which is then split as in the job splitting step. The type of data is analysed in the behaviour analysis step. At this step, there are three processes which take place. The first one is to test the type of jobs, the next is to monitor the data behaviour and then check the slave machine capability. Then the jobs will be allocated to each slave machine accordingly. Then each job will be completed by the corresponding slave machine and the result will be given back. Here the copy of the results will be placed in the slaves and the references will be kept in the master in a hash table. Using this, the results can be easily accessed for any future reference.

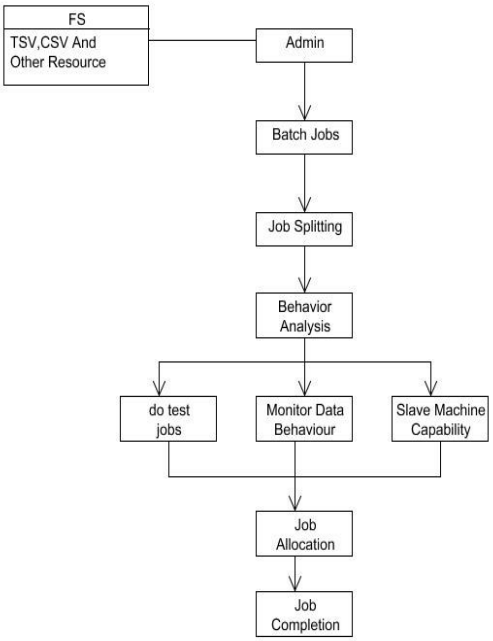

Fig 3 Proposed system Architecture

Block Chain Algorithm

It is an efficient algorithm based on dynamic data splitting strategy on Hadoop. Here the data which is to be processed is split dynamically on the basis of the capacity of the slave nodes on clusters, type of data which is to be processed. In order to determine the factors, sampling of data is carried out. Her a pre-run job instead of being prioritized for execution than the real jobs, the samples are being integrated along with the original tasks that are being scheduled. While processing, the real job, the intermediate job return their result quickly. Having these results, we can determine those before mentioned factors and the map tasks can be sent to those slaves that has the capacity to process the related map tasks effectively. Once the map tasks get processed, the keys of partition are stored on the index which is maintained in the master. Later, the reduce task starts, process and the value for the keys are stored in the index. The processed results those generated are stored locally in the memories of the slave nodes and a copy of it is also sent to the master. It is accessed based on the cache based rendering. Also,

a hash table, which is maintained in the master, is that whenever a user accesses an output, the result is searched and given to the user. It is cached and whenever the same output is searched, then the master directly goes to the specific slave where the result resides since it has a hash table that holds the key for all the results that has been accessed earlier. Hence, it significantly reduces the time complexity of the output.

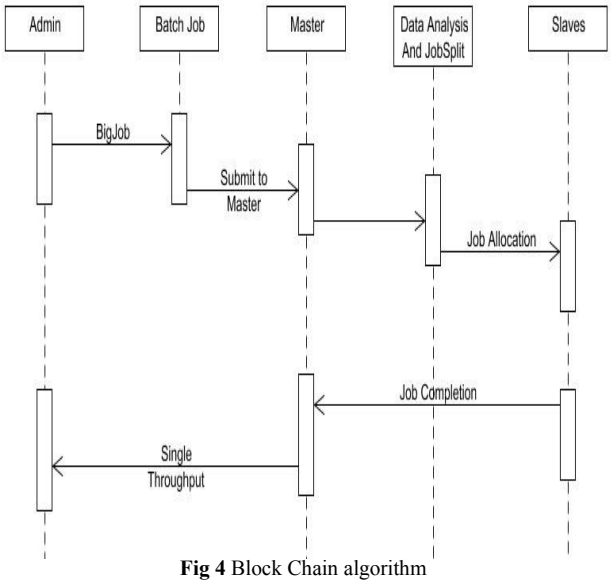

Fig 4 Block Chain algorithm

Web Crawling

Web crawling is nothing but the fetching of data Resources residing in any of the web server with or without the knowledge of the Resource provider. Generally the Resource servers will incur high traffic and load while web crawling is done and it can be detected by traditional DOS (Denial of Service) detection schemes, which will restrict the user to take the resources beyond the limit, when implemented through a single server. So we implement an efficient way to crawl resources residing any server through our Hadoop cluster in which the traditional DOS detection methods could fail to detect the attack. All the Resources crawled will be stored for future use.

System Modules

The proposed system can be developed into several system modules for easy implementation. The working modules that are comprised in this project are

1. Data Skew and its problem.
2. Web Crawling using Hadoop.
3. Data Analysis by Sampling and Data Skew Mitigation.
4. Query Processing using MapReduce and Block Chain (inclusive of live data migration)

Data Skew and Its Problem

The imbalance in the amount of data assigned to each task causes some tasks to take much longer to finish than others and can significantly impact performance. This paper presents LIBRA, a lightweight strategy to solve the data skew problem for reduce-side applications in Map Reduce. In Traditional Hadoop clusters the data is divided into partitions on a static way that each slave node will be allocated with equal size of data which might lead to the Data Skew problem. As all slave nodes are not equally capable and also the complexity of data will not also be similar some slave nodes may process the task assigned for a longer time than the other slave nodes in the cluster. Master node will look up for the completion of all Map Tasks and in this scenario will wait for the long running task on the slave node to complete. This time lag should be eliminated so that the master to through the overall output.

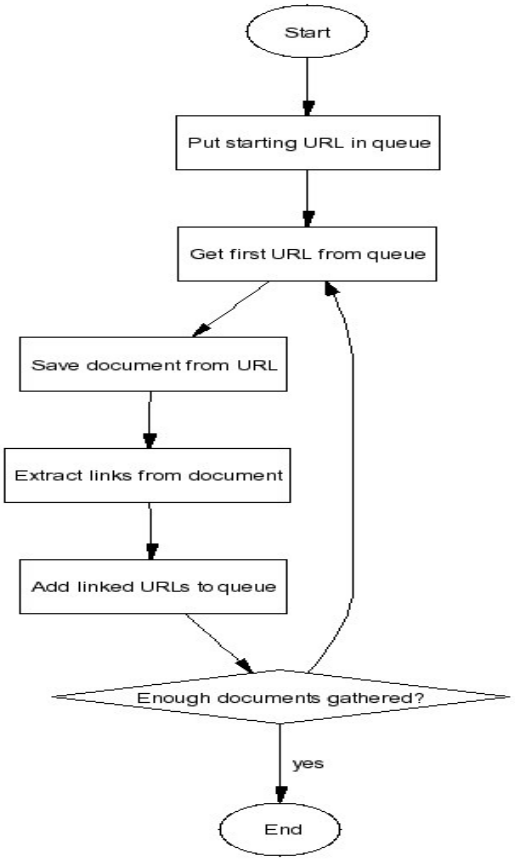

Fig 5 Block Diagram of Web Crawling

Web crawling using hadoop

Web crawling is the fetching of data Resources residing in any of the web server with or without the knowledge of the Resource provider. Generally the Resource servers will incur high traffic and load while web crawling is done and it can be detected by traditional DOS (Denial of Service) detection schemes when implemented through a single server.

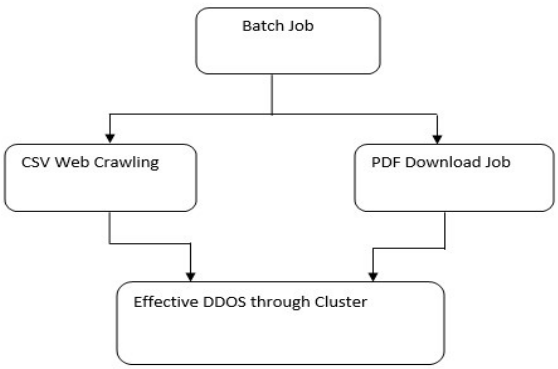

Fig 6 Process of Web Crawling in Hadoop

So we implement an efficient way to crawl resources residing any server through our Hadoop cluster in which the traditional DOS detection methods could fail to detect the attack. All the Resources crawled will be stored for future use. Web crawling jobs include PDF Web crawling stored as PDF format and medical question and answers web crawling stored as CSV (Comma Separated Values) format.

Data Analysis By Sampling And Data Skew Mitigation

Due to the dataskew problem which causes many difficulties that occur if the type of the input data is not known, it can be solved by examining the type of the data before deciding how to partition it. There are two common ways to solve this problem. One of the techniques is to launch some pre-run jobs that check the type of data, collect the distribution statistics and decide the correct partition of data. The drawback of this technique is that the real job process cannot be started till the pre-run jobs complete. This takes more time for the actual process to start. The Hadoop range partition belongs to this category. The other technique is to integrate the sampling of data along with the normal map reduce process and then generate the distribution statistics after all the map reduce task complete. Since the actual process cannot start till the partition of data is decided, this technique cannot take the advantage of parallel processing of data between map and reduce phases. For avoiding these complexities we take a different approach that the data is assigned earlier based upon the capacity of the machines, it is split dynamically rather than static splitting of data depending upon the complexities, job execution rate, etc. that was examined and then assigned to the slave nodes. As a result we show the job completion of slave nodes in an efficient and optimal manner.

Query Processing Using Mapreduce And Block Chain

The Query processing can be done using MapReduce Framework of Hadoop system and the results are rendered to the client page. Multiple Reduce tasks are done on the map task to give the reduced object and the results can be rendered as and when user needs. The response time is calculated for Querying with MapReduce. We also implement the same Querying with our own approach Block Chain Algorithm which stores the Map task output locally on the slave machine and uses cache based rendering of results. The Response time is compared with the Map Reduce response time and is up to the mark. This can show more effectiveness when used with large data with small cluster setup.

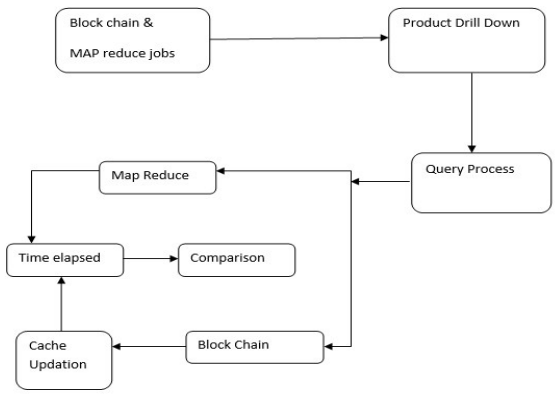

Fig 7 Query processing – Block Chain

Live Data Migration

Live data migration is a technique where it enhances the processing of the data on live action. In simple terms, it can be defined as in case of a slave node slowing down when a huge log of data is being processed, then to tackle this situation and to manage the performance of the slave node, a portion of data is sent to other slave node on the cluster which has a minimal job for processing or which may have high processing capacity. This reduces the dataskew to a considerable amount thus enhancing the overall throughput.

CONCLUSION

Hence we designed and developed a Hadoop cluster which can mitigate Dataskew problem using the innovative dynamic splitting algorithm called the Block Chain Algorithm and also we show its effectiveness by comparing the time complexities of Block Chain Algorithm and MapReduce Algorithm. We also demonstrate the effectiveness of the Hadoop cluster for web crawling in various real time web servers. Also we preserve the results valid even in the heterogeneous environment as the performance evaluation of the outputs in various situations are preferably unique and significant in the proposed system.

Acknowledgement

The work was supported in part by the " Vel
Tech High Tech Dr.RangarajanDr.Sakunthala Engineering College", affliated to "Anna University", Chennai, Tamil Nadu. The authors would like to thank the assistant professor Ms.RajaSaranyaKumari for her guidance and for the comments provided on an earlier draft of this paper. The authors would also like to thank the anonymous reviewers for their valuable comments and suggestions to improve the manuscript.

Reference

1. Qi Chen, Jinyu Yao, and Zhen Xiao, Senior Member, IEEE “LIBRA: Lightweight Data Skew Mitigation in MapReduce” IEEE Transactions on parallel and Distributed systems, VOL. 26, NO.9, september 2015
2. J. Dean and S. Ghemawat, “Map reduces: Simplified data processing On large clusters,” Commun. ACM, vol. 51, pp. 107–113, Jan.2008.
3. Apache hadoop [Online]. Available:http://lucene.apache.org/Hadoop/, 2013.
4. M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, “Dryad: Distributed Data-parallel programs from sequential building blocks,” In Proc. ACM SIGOPS/EuroSys Eur. Conf. Comput. Syst., 2007, pp. 59–72.
5. Y. Kwon, M. Balazinska, and B. Howe, “A study of skew in map reduce Applications,” in Proc. Open Cirrus Summit, 2011.
6. C. B. Walton, A. G. Dale, and R. M. Jenevein, “A taxonomy and Performance model of data skew effects in parallel joins,” in Proc.Int. Conf. Very Large Data Bases, 1991, pp. 537–548.
7. D. J. DeWitt, J. F. Naughton, D. A. Schneider, and S. Seshadri, “Practical skew handling in parallel joins,” in Proc. Int. Conf. Very Large Databases, 1992, pp. 27–40.

8. J. W. Stamos and H. C. Young, "A symmetric fragment and replicate Algorithm for distributed joins," IEEE Trans. Parallel Distrib.Syst., vol. 4, no. 12, pp. 1345–1354, 1993.
9. V. Poosala and Y. E. Ioannidis, "Estimation of query-result distribution And its application in parallel-join load balancing," in Proc.Int. Conf. Very Large Data Bases, 1996, pp. 448–459.
10. Y. Xu and P. Kostamaa, "Efficient outer join data skew handling in Parallel dbms," Proc. VLDB Endowment, vol. 2, no. 2, pp. 1390–1396, 2009.
11. S. Acharya, P. B. Gibbons, and V. Poosala, "Congressional samples for approximate answering of group-by queries," in Proc.ACM SIGMOD Int. Conf. Manage. Data, 2000, pp. 487–498.
12. A. Shatdal and J. F. Naughton, "Adaptive parallel aggregation algorithms," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 1995,pp. 104–114.
13. Y. Kwon, M. Balazinska, B. Howe, and J. Rolia, "Skew-resistant parallel processing of feature-extracting scientific user-defined functions," in Proc. ACM Symp. Cloud Comput., 2010, pp. 75–86.
14. S. Ibrahim, J. Hai, L. Lu, W. Song, H. Bingsheng, and Q. Li, "Leen: Locality/fairness-aware key partitioning for mapreduce in the cloud," in Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci., 2010, pp. 17–24.
15. G. Benjamin, A. Nikolaus, R. Angelika, and K. Alfons, "Handling data skew in mapreduce," in Proc. Int. Conf. Cloud Comput. Serv.Sci., 2011, pp. 574–583.
16. G. Benjamin, A. Nikolaus, R. Angelika, and K. Alfons, "Load balancing in mapreduce based on scalable cardinality estimates," in Proc. Int. Conf. Data Eng., 2012, pp. 522–533.
17. Y. Kwon, M. Balazinska, B. Howe, and J. Rolia, "Skewtune: Mitigating skew in mapreduce applications," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2012, pp. 25–36.
18. Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," IEEE Trans. Parallel Distrib. Syst.), vol. 24, no. 6, pp. 1107–1117, Jun.2013.

****

How to cite this article:

RajaSaranyaKumari R., Lenny Arul Jegan and Yuvasree J.2016, Mitigation of Data Skew Using Block Chain Algorithm. *Int J Recent Sci Res.* 7(5), pp. 11306-11311.

TSSN 0976-3031