IJCTA, 9(27), 2016, pp. 423-430
ISSN: 0974-5572

EDSHA: An Efficient Data Skew Handling Approach For Mapreduce Model Using Time Series Data

B. Arputhamary*, S. Dhanalakshmi and L. Arockiam*****

ABSTRACT

In the epoch of Big Data, wide ranging sources produce a huge volume of structured and unstructured data. It is difficult to process the Big Data and it requires a large number of systems to work in parallel. MapReduce is a parallel scale programming model which is useful to handle huge amount of data in parallel. In heterogeneous environment, the nodes are not similar. So, there is a chance that the high performance nodes can process the data faster and may take less time to complete than other nodes. In MapReduce model, data skew arises due to the uneven distribution of data to the nodes and can occur either at the Map or Reduce phase. This causes the skewness of data hence the whole task is getting delayed by the slowest processing task which is known as 'straggler' and it keeps some of the machines idle for a long time until others completes its task. In order to accomplish this, the proposed model introduces EDSHA (Efficient Data Skew Handling Approach) as a new phase in between the Map and the Reduce phases which assigns the high end node with huge data dynamically in their order of efficiency in terms of execution time and memory size. Moreover the prediction for an additional year is obtained in this model using sales data.

Keywords: Big Data, Data Skew, MapReduce, Prediction, Straggler.

1. INTRODUCTION

In recent years Big Data has become a buzzword. Big Data termed as 5 V's – high level of Volume, Velocity, Variety, Veracity and Value. The network bandwidth increased by generating huge volumes of data produced by various companies, researches and governments. Data can come from anywhere; either structured or unstructured, as sales statistics, financial balances, cell phone signals, business deals and climate data which are embedded in electronic devices as Global Positioning System (GPS) devices with sensors, consumer feedback and comments from various websites, posting a text, image, audio and video on social networking websites. To access the numerous popular applications larger data sets can be used and the data scale ranging from bits to Yotta bytes. To process such amount of data in a single node will be time consuming. The data itself may be too large to store in a single node. Due to the issue of time complexity, the data can be processed initially and then stored in separate storage space and the workload can be distributed among more than one system. Google introduced the Google File System (GFS), a disseminated file system model for huge range data processing with MapReduce programming model [6]. With the help of the MapReduce model 20 Peta bytes of data are processed by Google every day.

MapReduce is the kernel of the Apache's Hadoop. In MapReduce, the performance and scalability can be increased because the huge data set can be split into many smaller partitions and the task is partitioned into many smaller tasks. These tasks run on the many dissimilar nodes in a cluster. Various companies like eBay, Facebook, IBM, LinkedIn and Twitter use Hadoop to handle huge quantities of records [6]. To balance the cargo, Hadoop distributes the data based on existing disk spaces to the nodes in the cluster. In heterogeneous backdrop, the

* Research Scholar, Department of Computer Science, Mother Teresa Women's University, Kodaikanal, India, *Email: arputhambaskaran@rediffmail.com*

** Research Scholar, Department of Computer Science, Bishop Heber College, Tiruchirappalli, India, *Email: dhanas.gopal@gmail.com*

*** Associate Professor, Department of Computer Applications, St. Joseph's College, Tiruchirappalli, India, *Email: larockiam@gmail.com*

straggler can occur only because of the nodes that are not similar and may differ in the computing and disk capability.

Existing models focus on the tasks of high performance nodes, which may take less time to complete and can process the data faster than the low performance nodes. This causes the skewness of data; hence the whole job gets delayed by the slowest processing task which is known as straggler [7]. For that data skew cannot be rescued by simply transmitting the task to the other node. The existing techniques for mitigating data skew issues do not attempt to reduce the skew rather it specifies the re-partitioning. It also leaves out the processing time and energy costs that agree the implementation of existing algorithms.

1.1. Data Skew

Data skew mainly refers to a non-uniform allocation across partitions in a data set. An irregular allocation degrades the performance of the overall implementation as the CPU is idle and waits for some other nodes to complete the task with superior quantity. Skewed distributions commonly follow any one of the general distributions such as Zipfian, Gaussian and Poisson. In a statistical distribution, skewness is termed as asymmetry i.e. unevenness of data and its value can be positive or right, negative or left and evenly distributed. In MapReduce, skew happens when one node has more data assigned to be processed than others either of the Map or Reduce phase [11]:

Mapper Side: Following are some of the reasons for the data skew to occur at mapper side [7], [9]:

1.1.1. Records are Expensive

Map task processes a set of data as key/value pairs, one at a time. Hence, the processing time does not vary and it requires more CPU and memory to process. It shows that expensive records may be larger than other records. PageRank is an algorithm that experiences the Map phase, which assigns ranks (weights) to each and every apex in a graph by aggregating ranks of its inbound neighbor.

1.1.2. Heterogeneous Map

MapReduce can be used to follow an n-ary operation by plausibly concatenating different data sets as a unique input. Each data set may need discrete processing which leads to multi-modal distribution of job at run time.

Reducer Side: Various causes of the data skew issues at reducer side are as follows [9]:

1.1.3. Data Partitioning

The range partitioning is mostly used to partition the data sets. Data partitioning can be done either in two ways using hash functions or range keys which yields an uneven partition and finally resulting in failures. Some reducers may get the more data than the others. The partitioning logic must not rely on the tuples hence it reflects a skew at the reducer side.

1.1.4. Key Groups are Expensive

Reducer task processes the (key/value) pair. Mapper task processes the expensive records at run time, but Reducer task causes the skew on expensive key groups.

2. RELATED WORKS

Q. Chen et al., [1] proposed LIBRA model which avoids execution of pre-run sample tasks in order to decide the distribution of intermediate data. It does sampling during the normal execution of map tasks only. Reducer can immediately start consuming the data as soon as this starting map tasks get completed. LIBRA also partitions the keys which are large in size and also consider the heterogeneous nodes while executing this policy. Y. Kwon et al.,

[2] proposed SkewTune to identify the idle nodes. Next it offloads data on straggler task. It considers the highest expected running time as metric by using the efficient partitioning scheme. But the model is idealistic and not practical. Q. Chen et al., [3] proposed a scalable model for data skew called straggler. This model uses process bandwidth and progress rate in a current phase to decide slow tasks and it calculates tasks remaining time and makes predictions about process speed using EWMA (Exponentially Weighted Moving Average).

FAN Yuanquan et al., [4] described mitigation at both the mapper and the reducer side which is the main motivation behind this work. In a heterogeneous environment to improve the performance of reducer phase, load balancing approach is helpful. Ultimately make the span of MapReduce jobs, increase load during the Map phase. K. Slagter et al., [6] provided the partitioning algorithm that improves load balancing and memory utilization. It is done through superior sampling algorithm and partitioner. Tera Sort mechanism is employed to evaluate the proposed algorithm. Nawab Wajid et al., [7] suggested the different type of skew that arises in MapReduce and the techniques to mitigate the skew issues. Y. C. Kwon et al., [8] presented a detailed study of skew at run time in MapReduce environment and surveyed familiar sources of skew in MapReduce applications and established skew problems using real workloads.

Y. C. Kwon et al., [9] implemented an automatic for user-defined MapReduce programs and skew mitigation approach. SkewTune identifies the task with the longest remaining running time. The unrefined input data of this straggling task are then re partitioned in a way that fully consumes the nodes in the cluster and conserves the ordering of the input data so that the desired output can be reconstructed by clustering. Benjamin Gufler et al., [10] presented an algorithm to estimate the cost and proposed balancing approaches as fine partitioning and dynamic fragmentation are based on the cost model and focused with both skewed data and complex Reducer tasks. J. Vinutha et al., [11] explored different types of skew and the techniques to mitigate the data skew issues.

V. A. Nawale et al., [12] presented various methodologies and techniques to minimize skew for the MapReduce and described the advantages and limitations of those approaches in partitioning. Y. C. Kwon et al., [14] proposed Skew Reduce, a system that statically optimizes the records partitioning according to user defined cost functions.

3. PROPOSED SYSTEM MODEL

In the proposed approach EDSHA (Efficient Data Skew Handling Approach), huge volume of online sales data are collected and the data is distributed month wise to 12 partitions to the nodes in their order of efficiency in terms of execution time and memory size which preprocesses the data. MapReduce systems have become popular for processing large data sets and are increasingly being used in online applications. In contrast to simple application scenarios like word count, other applications involve complex computations which pose new challenges to MapReduce systems. In particular, (a) the processing time complexity of the reducer task is typically high, and (b) online data is often skewed. This leads to highly varying execution times result in low resource utilization. The two phases of MapReduce programming are the Map and the Reduce phase. The input data from each partitions are assigned to the mapper and the mapper, split the input data into M splits using (key, value) pair. In the MapReduce model, data skew arises due to the uneven distribution of data either at the Map or Reduce phase. Data skew is amplified when Reducers have highly varying execution times which cause the Reducers with a low load to wait for the reducers with high load. The problem of data skew issue is solved by using the traditional approaches as load balancing and re-partitioning which leads to degrades the performance and reduces the idle time of the processors. In this proposed model, range partitioning is used in order to split the data month wise. Data are partitioned based on range keys. The sales may vary among these partitions seasonally or occasionally. Mapper sorts and shuffles the data; while Reducer partitions it. The proposed EDSHA phase assigns the high end node with huge data dynamically and does the prediction for future. In Fig. 1 split0, split1, …., split11 (12 mappers) from each split is shuffled and sorted. Multiple mappers are assigned with multiple reducers by using custom partitioner. To identify the high end node, Fig. 2 depicts that the nodes are identified in their order of execution time from the least to the highest. So the reducers are re-arranged and the high end node is shown. Then the Reducer merges the data. But for Time Series

data the traditional approach is not helpful to solve the issue. Because sales data may vary based on the seasonality it degrades the overall performance. To overcome this, EDSHA is proposed to assign the high end node with huge data dynamically. In addition, it predicts the month with the highest retailing detail.

Algorithm – EDSHA (Efficient Data Skew Handling Approach):

Step 1: Read the input online sales data

Step 2: Map phase (Partitions according to the situation)

Step 3: Shuffling and Sorting

Step 4: EDSHA phase: Let R be the set of reducers with different computing capability and M be the mappers which contains the set of all data in each month which has different sizes

$M \leftarrow \{M_1, M_2, M_3, ..., M_n\}$ $R \leftarrow \{R_1, R_2, R_{3,...,} R_n\}$

Assigns the high end node with huge data dynamically based on the execution time and memory size

Step 5: Reduce phase (Merge the result)

Step 6: Predict the sales for subsequent months iteratively

Step 7: Visualize the result.

3.1. Configuration and Setup

EDSHA is proposed to handle the data skew that may arise while processing online sales data and predicts the sales for the upcoming months.

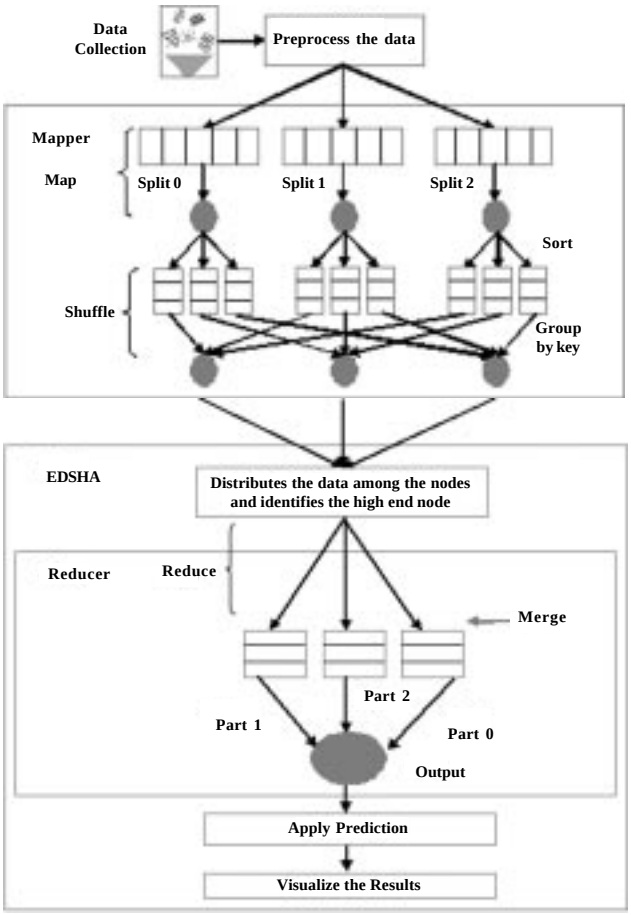

Figure 1: Proposed Methodology of EDSHA

4. EVALUATIONS AND DISCUSSIONS

An Efficient Data Skew Handling Approach (EDSHA) is maintenance free and takes low overheads. A quantitative parameter to express the accuracy rate and prediction provides a framework to compare and improve different mapping techniques. It assigns the high end node - the one which has the least execution time i.e. the highest accuracy rate - with huge data and also makes prediction using online sales data for the upcoming months. The essence of the proposed method is that - The data imbalance is handled effectively by using EDSHA by identifying the most efficient node in terms of minimal execution time as the high end node. The principle of data skew handling by avoiding straggler or idle times by using EDSHA algorithm effectively operates in the heterogeneous environment. Consumption of time is reduced.

Finding Out the High End Node:

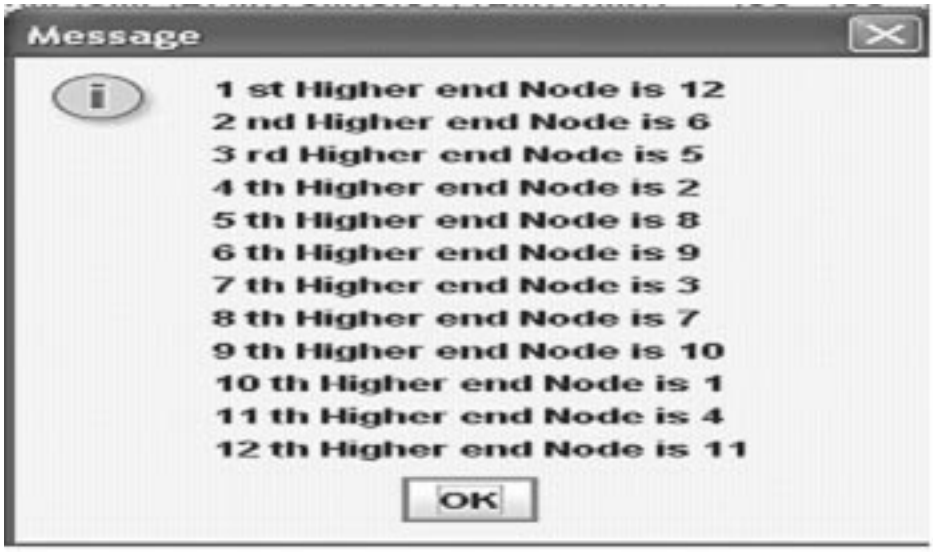

Figure 2: Representation of high end node with least execution time in their order

Accuracy of the Existing Vs Proposed Method:

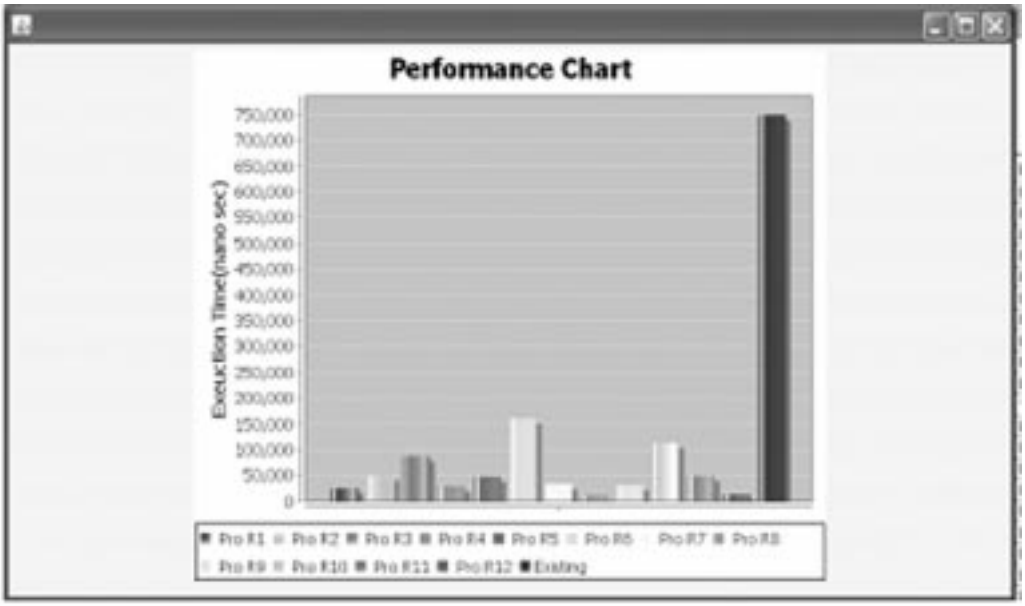

Figure 3: Calculating accuracy of existing and proposed techniques in data skew handling

Table 1
Comparison Analysis of Execution Time of Skew Handling

Mappers	*Before Skew Handling*		*After Skew Handling*	
	Reducers	*Exec Time in NS*	*Reducers*	*Exec Time in NS*
M1	R1	8594	R12	1594
M2	R2	4563	R11	2750
M3	R3	4391	R10	3797
M4	R4	4281	R9	1860
M5	R5	2984	R6	2781
M6	R6	2860	R5	2891
M7	R7	3032	R7	2906
M8	R8	2859	R4	4079
M9	R9	1969	R2	4484
M10	R10	3875	R8	2765
M11	R11	2844	R3	4219
M12	R12	1672	R1	8313

It is evident that the proposed method gives the highest accuracy rate i.e. least execution time. Figure 3 represents the execution time taken by the proposed system is relatively less than the existing method. So in terms of accuracy in execution time, the proposed method overrides the existing technique.

M1 …. M12 are the Mappers and R1 …. R12 are the Reducers which are used to assign the high end node based on their execution time in nanoseconds (NS) in Table 1. In MapReduce phase, the Mappers are assigned with the Reducers in parallel before skew is taken into account; whereas the Mappers are assigned with the Reducers according to the high end node in EDSHA phase.

Analysis of Execution Time of Skew Handling:

The execution time taken after skew handling is found to be better than the earlier as given in Figure 4. Execution time of skew handling is mentioned in nanoseconds.

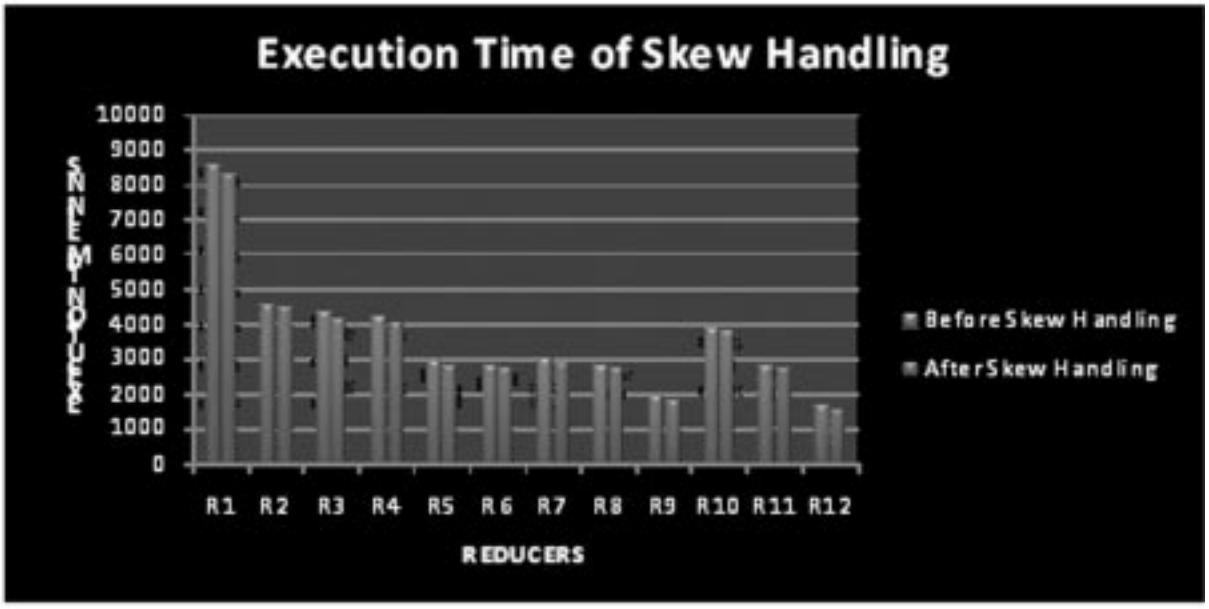

Figure 4: Representation of execution time of skew handling

Monthly Prediction of Sales for Next Year:

The monthly prediction value and threshold value achieved is illustrated in Figure 5 by using the online sales data. The actual data is marked in red and the predicted data is marked in blue. It is clear that the predicted data is higher than the actual data.

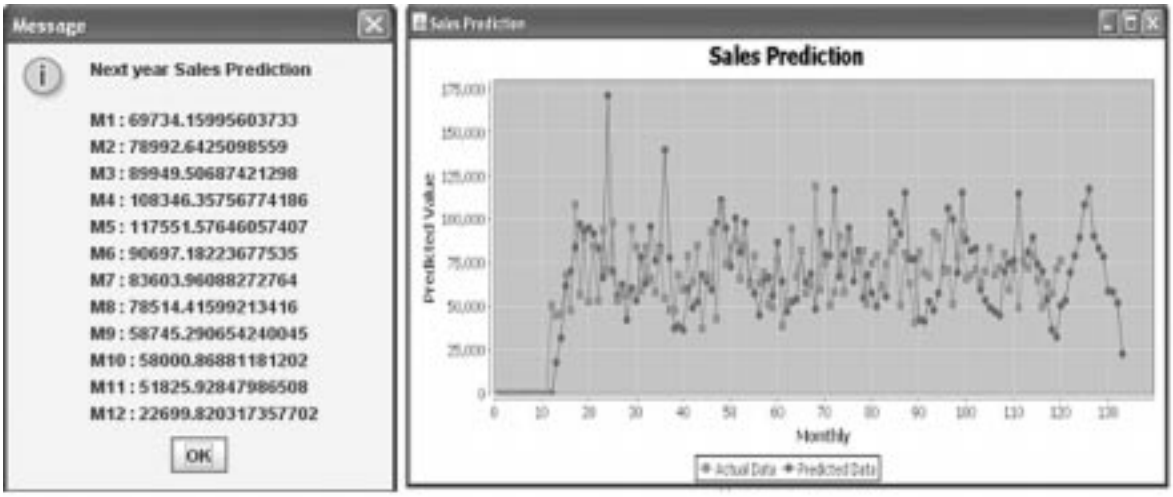

Figure 5: Representation of prediction using Holt Winters Method

5. CONCLUSION

In a business environment, prediction analysis plays a vital role at the proper time to improve their incremental status with low cost. At the same time, data skew is an important issue caused by straggler. In this paper an Efficient Data Skew Handling Approach (EDSHA) is proposed to estimate the accuracy and it can respond automatically and incrementally update the output according to the changes in the input. Data locality is measured as one of the main optimization means for scheduling. It is implemented based on Hadoop, compatible with the original prediction interfaces and transparent to users. The proposed method overcomes the straggler node and reduces the idle time of the processor. In a heterogeneous environment, it assigns the high end node with huge data to complete within a specified time. Hence, it is inferred that the proposed method is better in accuracy (in terms of memory size and processing time) and execution time needed, and is blatantly advantageous over the existing ones.

REFERENCES

[1] Q. Chen, J. Yao, and Z. Xiao, "LIBRA: Light Weight Data Skew Mitigation in MapReduce", *Parallel and Distributed Systems*, IEEE, **99**, 2014.

[2] Y. Kwon, M. Balazinska, B. Howe and J. Rolia, "SkewTune: Mitigating Skew in MapReduce Applications", *In Proc. of the ACM SIGMOD International Conference on Management of Data*, 2012.

[3] Q. Chen, C. Liu and Z. Xiao, "Improving MapReduce Performance Using Smart Speculative Execution Strategy", *IEEE Transactions on Computers (TC)*, **63(4)**, 2014.

[4] FAN Yuanquan, WU Weiguo, XU Yunlong and CHEN Heng, "Improving MapReduce Performance by Balancing Skewed Loads", 2014.

[5] D. DeWitt, J. Naughton, D. Schneider and S. Seshadri, "Practical Skew Handling in Parallel Joins", *In VLDB*, 1992.

[6] K. Slagter, Y. Chung, D. Zhang and C.H. Hsu, "An Improved Mechanism for Optimizing Massive Data Analysis Using MapReduce", *The Journal of Supercomputing*, **66(1)**, 539-555, Springer, 2013.

[7] Nawab Wajid, S. Satish and T.N. Manjunath, "A Survey on Hadoop MapReduce Framework and the Data Skew Issues", *International Journal of Scientific Research Engineering & Technology*, ISSN 2278-0882, **4(4)**, 2015.

[8] Y. Kwon, M. Balazinska, B. Howe and J. Rolia, "A Study of Skew in MapReduce Applications", *In The 5th International Open Cirrus Summit*, 2011.

[9] Y. Kwon, M. Balazinska, B. Howe and J. Rolia, "SkewTune: Mitigating Skew in MapReduce Applications", *In Proc. of the ACM SIGMOD International Conference on Management Data*, 25-36, 2012.

[10] B. Gufler, A. Kemper, A. Reiser and N. Augsten, "Handling Data Skew in MapReduce", *International Conference on Cloud Computing and Services Science*, CLOSER 2011.

[11] J. Vinutha and R. Chandramma, "Skew Types Mitigating Techniques to Increase the Performance of MapReduce Applications", *International Journal of Emerging Technology and Advanced Engineering*, ISSN 2250-2459(Online), **5(2)**, 2015.

[12] V.A. Nawale and P. Deshpande, "Survey on Load Balancing and Data Skew Mitigation in MapReduce Applications", *International Journal of Cpmputer Engineering & Technology*, ISSN 0976-6367(Print), ISSN 0976-6375(Online), **6(1)**, 32-41, 2015.

[13] S. Sridhar, S. Choudary, N. Raigur, D. Kumar and S. Kumar, "Dynamic Mitigation of Data Skew in MapReduce", *SS International Journal of Multidisciplinary Research*, ISSN 2395-7964(Online), **2(4)**, 2016.

[14] Y. Kwon, M. Balazinska, B. Howe and J. Rolia, "Skew-Resistant Parallel processing of Feature-Extracting Scientific User Defined Functions", *In Proc. of the First SOCC Conf*., 75-86, 2010.

[15] B. Gufler, A. Kemper, A. Reiser and N. Augsten, "Load Balancing in MapReduce Baased on Scalable Cardinality Estimates", *In Proc. of the 28th ICDE Conf*., 522-533, 2012.

[16] G. Ananthanarayanan, I. Stoica, A. Greenberg and S. Kandula, Y. Lu, E. Harris and B. Saha, "Reining in the Outliers in Map-Reduce Clusters using Mantri", *In Proc. of the 9th OSDI Symp*., 2010.

[17] Y. Xu, "Handling Data Skew in Parallel Joins in Shared-Nothing Systems", *In Proc. of the SIGMOD Conf*., 2008.

[18] Y. Xu and P. Kostamma, "Efficient Outer Join Data Skew Handling in Parallel DBMS", *In VLDB*, 2009.

[19] Hadoop. http://hadoop.apache.org/.

[20] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", *In Proc. of the 6th OSDI Symp*., 2004.

[21] J. Lin, "The Curse of Zipf and Limits to Parallelization: A Look at the Stragglers Problem in MapReduce", *In 7th Workshop on Large-Scale Distributed Systems for Information Retrieval*, 2009.

[22] Y. Kwon, M. Balazinska, K. Ren, B. Howe and J. Rolia, "Managing Skew in Hadoop", *IEEE Data Eng. Bull.*, **36(1)**, 24-33, 2013.

[23] Z. Xiao, W. Song and Q. Chen, "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment", *IEEE Transactions on Parallel And Distributed Systems*, **24(6)**, 1107-1117, 2013.

[24] Custom Partitioner. https://hadooptutorial.wikispaces.com/Custom+partitioner.

[25] S. Dhanalakshmi, B. Arputhamary and Dr. L. Arockiam, "A Survey on Data Skew Mitigating Techniques in Hadoop MapReduce Framework", *International Journal of Computer Science and Mobile Computing*, **5(7)**, 2016.