A Comparative Investigation of Sample Versus Normal Map for Effective BigData Processing

Shyavappa Yalawar, V. Suma and Jawahar Rao

Abstract MapReduce is an effective tool for the parallel-processing of data. A major problem in practice, MapReduce Skew of the data: imbalance amount of data for each task consigned. Because some of the tasks to last much longer than other, and can greatly affect performance. A scale that is lightweight strategy for data skew problem solving Applications, to the reducer side in MapReduce. In contrast to previous work scale is no need to scan in front of a series of input data or to prevent the overlap between the maps and reduce phases. System uses innovative idea take samples, which can achieve a high level of calculation and produce accurate approximation for the distribution of intermediate data by scanning only a small portion of the data on intermediate Map of the normal processing. It allows the reduction of tasks, to start the copying once the sample map functions selected (only a small part Map of tasks that have been fully spent for the first time). It supports split large clusters make connotations when applied and the total Output data is set up. System is implemented in Hadoop and Our experiments show that the implementation of some popular applications to speed up on negligible and can speed up Factor 4.

Keywords Bigdata · Mapreduce · Dataskew · Straggler

S. Yalawar (✉)
Post Graduate Programme, Computer Science and Engineering, Department of Information Science and Engineering, Dayananda Sagar College of Engineering, Bangalore, India
e-mail: shyavappayalawar@rediffmail.com

V. Suma
Department of Information Science and Engineering, Research and Industry Incubation Centre (RIIC), Dayananda Sagar College of Engineering, Bangalore, India
e-mail: sumavdsce@gmail.com

J. Rao
Department of Industrial Engineering and Management,
Dayananda Sagar College of Engineering, Bangalore, India
e-mail: jawahar_rao@yahoo.com

© Springer India 2016
S.C. Satapathy et al. (eds.), *Information Systems Design and Intelligent Applications*, Advances in Intelligent Systems and Computing 434,
DOI 10.1007/978-81-322-2752-6_74

1 Introduction

For every two years data is doubling up because of Unstructured data are simply data that does not suitable into traditional relational database systems; the term Unstructured Data includes log files, emails, word documents, multimedia, video, PDF files, excel sheets, messaging text, images and graphics, graphs, charts, GPS records, and social media data which contains all the other elements on a huge scale. This huge collection of the data is nothing but a big data [1–13].

MapReduce has verified that one as an effective tool in order to process such huge data sets. There are various computing frameworks to process this data. Ex: Google Map Reduce, Microsoft Dryad and Apache hadoop. MapReduce has proven to be an effective tool to process this data. Apache hadoop is an open source and is broadly used. In a MapReduce, the job finishing point is decided by the slowest running task known as straggler. It has ability to stoppage the progress of the entire job. Data skew is the important task in MapReduce, to avoid the straggler problem data skew should perform well. Data skew means dividing the complete task or job equally to all the nodes.

MapReduce job completion time is still dependent on the work of the slowest running tasks. When a task consumes more time when compared to other (so-called backward) ingesting, it has the ability to sluggish down the progress of the entire job. Later in the image can for various reasons, including data skew is important soon.

The main reason relates to skew the data recorded in the real time are often distorted; User don't know the circulation of the expected data. It should be noted that this problem will not be speculative in MapReduce implementation of strategies for solving. Data migration is not a new problem related MapReduce Some literature studies at the beginning of parallel databases, but only to join and aggregation operations. While some of them after use of MapReduce, Developer have to develop their own data offset remission in most cases, more specific application. Hadoop map Reduce implementation of the standard hash function using static branch out of the intermediate data.

We create a procedure for taking samples of the new data sets of bigdata general knowledge. This method with a high degree of parallelism and little effort, cannot be attained a good approximation for the circulation of the data set temporarily.

We use an innovative approach for move between tasks, and the reduction of Split of large badges for compensation if they support the approval of the application semantics. In addition, the use of our method LIBRA each reducer process about the same amount of data.

Although the predictable of the simple heterogeneous computing platform that can be expand the distribution of the workload on the basis of the enhanced data accordingly, provide the best performance in the absence of Data skew.

LIBRA in Hadoop and assess its performance to some common applications. The experimental results have a clear picture; Libra can improve by up to a factor of 4 time carry out the task.

2 Proposed System

In this part of the proposed method that has the ability to approach system, and to achieve a solution Data skew for general applications. The map Reduce framework by us to adopt to implemented system is in Hadoop—1.0.0. The design goals listed below:

- *Total order*
- *The separation of a enormous data*
- *Transparency*
- *Parallelism*
- *Accuracy.*

2.1 System Overview

System architecture consists of HDFS, Map phase, Master node and Reduce phase.

- *Loading the input into the HDFS*: The HDFS is the distributed file system for hadoop which acts as a storage device. It contains Namenode, Datanode and secondary namenode before beginning to the data process it divides the large input data block into the small blocks so that the namenode assigns them to process the data and it saves the replica of processing data, in case the namenode goes down or offline the secondary namenode will start processing so that this is main advantage of the hadoop, that is the data will not miss in any point.
- *Mapper phase*: The mapper phase consists of two stages one is sample map and other is normal map, the system considering the 20 % of sampling data, according to the requirement we can change the percentage of the sample. Map phase is used to filtering and sorting of the data than it gives its output to the master node.
- *Master node*: The master node takes the output of the map phase then it decides and assigns the job to the reducer which block is to going to which reducer all these things can be handled by master node. In this system the master node only considering the 20 % of the data as a sample and according its output it is estimating data distribution for all other remaining data to process then it elect the partition and notifies to the worker node to process information. By this the map phase completes all the data processing without any overhead.
- *Reducer phase*: Now the output of mapper is goes to reducer it does the summary operation for sorting and shuffling the generated output by both phase and finally sends the output to the HDFS (Fig. 1).

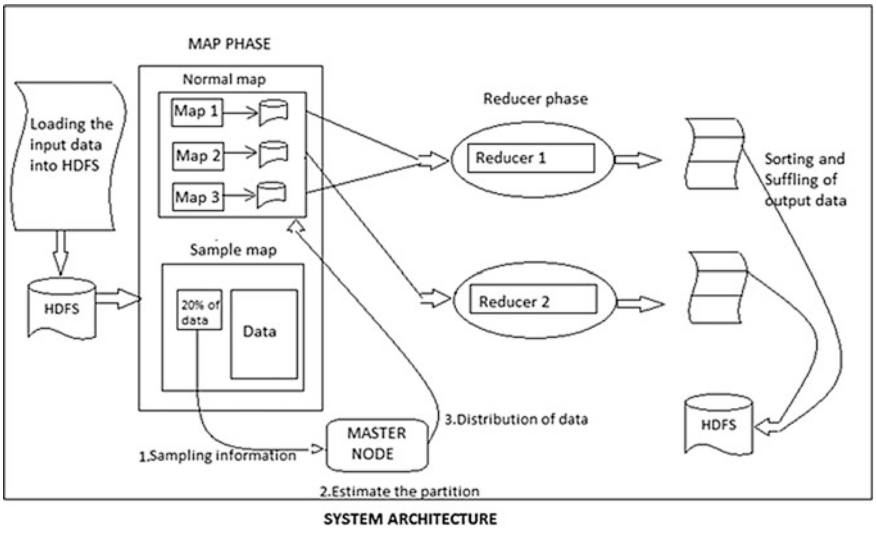

Fig. 1 System architecture represents the complete work flow

2.2 *Dividing the Large Cluster*

In a MapReduce framework the whole cluster appears in a single reducer to process the data due to this the data skew problem may occur. To avoid the particular data skew on the reducer side, the system splitting the large cluster. As shown in the above figure: example of large cluster splitting, the intermediate data taken from the network traffic analysis data set the name of the country and the count of the network devices is pair in the above figure. There are three countries Finland, Argentina and Singapore and the counts are 10,000, 4000 and 2000 respectively. Network traffic analyser is a data set in my project I have taken only 2 GB of the data (Fig. 2).

To solve the data skew on the reducer side we are trying split the large cluster into two so that both the reducers use same amount of workload. By dividing the count equally all the tasks can process at the same time both the reducer process the complete count by parallel. Another method to process the data is broadcast join but compare to this joining technique the cluster splitting is gives better result. By using the large cluster splitting in the above example the 60 % of the large data is optimized to the reducer 1 and the rest of reducer 2 data. By using this strategy lager cluster splitting it provides more flexibility in data skew mitigation.

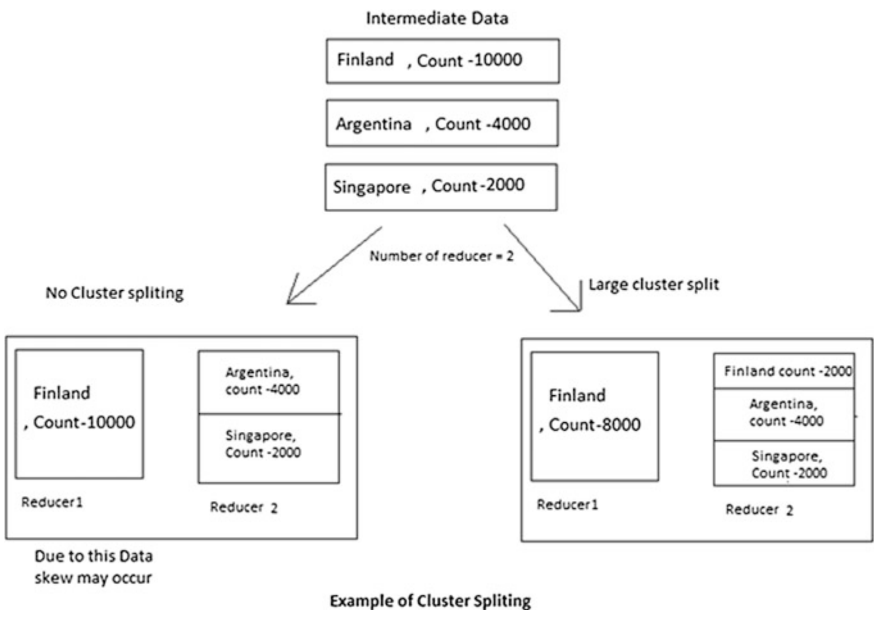

Fig. 2 Example of cluster splitting by using the network traffic analyzer data

3 Result

There are types of partitions in hadoop for data processing we are considering some of them—Key, Hash and Binary. The hash partition is default one to process the data. So we are processing our data set with all the three types of partitions. The above graph represents the job execution time with respect to the partitions. Key partition takes more time than the other strategy. By observing the above graph we can say that hashing technique better technique to process the data (Figs. 3 and 4).

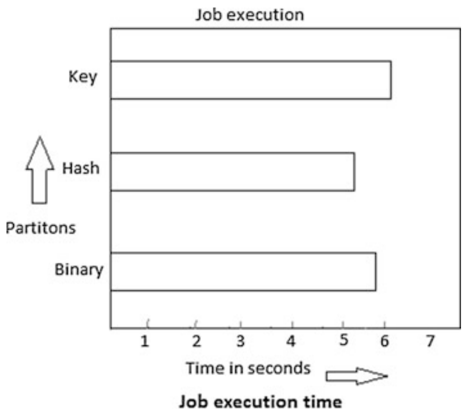

Fig. 3 Result of the different partitioner job execution time

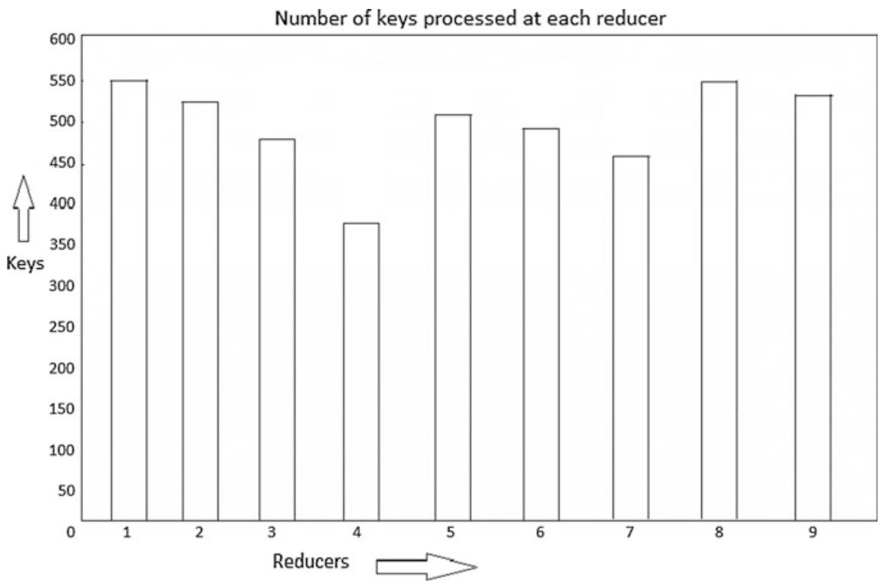

Fig. 4 Represents the 9 reducers taken as boundary versus number keys processed by the each reducer using three partition. Each *block* above denotes the average of the three partitions

4 Conclusion

Data skew mitigation is important in improving MapReduce performance. The technique LIBRA has proposed which implements a set of innovative skew mitigation strategies in an existing MapReduce system. This paper was presented a system; Where the input data is loaded to the HDFS and then map phase will execute the master node collects the information of the sample map through that information the master node estimate the data distribution for overall data by this strategy overhead is reduced and data skew on the reducer side is also solved. A unique feature of this paper is it supports for a large block and cluster split—which is adjusted to the Heterogeneous environments.

References

1. J. Dean and S. Ghema “Mapreduce: simplified data processing on large clusters” Communication acm volume 51, Jan 2008.
2. “Apache Hadoop, http://lucene.apache.org/hadoop/”.
3. M. Isard, Y. Yu, and D. Fetterly “Dryad” European conference on Compter Systems 2007.
4. Y. Kwon, M. Balazinka and Howe “A study of skew in MR Applications” Cirrus 2011.

5. C. B. Walton, A.G. Dale, and R. M. Jenevein, “A Taxonomy and Performance model of data skew effects in parallel joins”. International Conference on very large databases (VLDB), 1991.
6. D. DeWitt, J. Naughton, A. Schneider, and S. Seshadri, “Practical Skew Handling in Parallel Joins”, VLDB, 1992.
7. J. Stamoas and C. Young “Symmetric Fragment and Replicate algorithm for Distributed Joins”, IEEE TPDS Vol. 4 1993.
8. V. Poosala and Y. Ioannnidis, “Estimation of query-result distribution and its application in parallel-join load”, VLDB, 1996.
9. Y. Xu and P. Kostamaa. “Efficient outer join data skew handling in parallel DBMS”, VLDB vol. 2.2 2009.
10. S. Acharya, P. B. Gibbons and V. Poosala, “Congressional Samples for Approximate Answering of Group-by Queries”, International Conference on Mgmt of Data 2000.
11. A. Shatdal and J. Naughton, “Adaptive Parallel Aggregation Algorithms”, ACM Sigmod International conference on Mgmt of Data 1995.
12. J. Rolia and B. Howe, “Skew–Resistant Parallel Processing of Feature–Exatracting Scientific User-Defined Functions”, ACM Symposium on Cloud Computing, 2010.
13. Qi Chen Yao and Zhen Xiao “LIBRA: Lightweight data skew mitigation in mapreduce”, IEEE Transactions on parallel and distributed systems.