THOMSON REUTERS
ENDNOTE

[Veerala, 5(7): July 2018]
DOI: 10 10.5281/zenodo.1310778

ISSN 2348 – 8034
Impact Factor- 4.022

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

ENHANCING THE SPEED OF BIG DATA PROCESSING BY IMPLEMENTING MULTIPLE REDUCERS IN NETWORK

Bala Krishna Veerala, Professor

ABSTRACT

The main aim of this paper is to reduce the burden of the single reducer per map. Now-a-day, MapReduce performance improvement is very significant for big data processing. In previous works, we tried to reduce the network traffic cost for MapReduce job by implementing a novel data partition and aggregation scheme and also reduce the burden of the reducer while data processing. And also we have several schemes to provide an efficient and traffic-aware MapReduce scheme. But in these schemes we used one mapper and one reducer to process big data, then the reducer facing the overload problem in the network. Hence, in this paper we are implementing multiple reducers for single map instead of single reducer. This proposed implementation can balance the load of the reducer by load balancing mechanism.

Keywords*: MapReduce, Network, Load balancing, Mapper, Reducer.*

I. INTRODUCTION

Processing huge-scale datasets has grow to be an more and more essential and hard hassle as the amount of data created by using on line social networks, healthcare enterprise, scientific studies, etc., explodes. MapReduce is a simple but powerful framework for processing large-scale datasets in a allotted and parallel style.

A manufacturing MapReduce cluster may also consist of tens of lots of machines. The saved data are commonly prepared on allotted report systems (e.G., Google File System (GFS), Hadoop Distributed File System (HDFS), which divide a big dataset into records chunks and keep a couple of replicas of each chunk on specific machines. A data processing request underneath the MapReduce framework, referred to as a task, includes two sorts of tasks: map and decrease. A map venture reads one data bit and processes it to supply intermediate results (key-fee pairs). Then lessen tasks fetch the intermediate results and carry out in addition computations to produce the very last result.

Map and reduce tasks are assigned to the machines inside the computing cluster by means of a master node which maintains song of the reputation of these obligations to control the computation procedure. In assigning map responsibilities, a crucial attention is to location map obligations on or close to machines that shop the input statistics chunks, a trouble referred to as statistics locality. For every challenge, we call a machine a local system for the project if the records bite related to the venture is stored locally, and we name this venture a local project at the gadget; otherwise, the gadget is known as a far off system for the undertaking and correspondingly this undertaking is known as a far off challenge at the machine. Locality also refers back to the fraction of responsibilities that run on local machines. Improving locality can lessen both the processing time of map duties and the community site visitor's load since fewer map duties need to fetch information remotely. However, assigning all tasks to nearby machines may lead to an uneven distribution of tasks among machines, i.e., some machines be able to be heavily congested while others may be idle. Therefore, we need to strike the proper balance between information-locality and load-balancing in MapReduce. We name the set of rules that assigns map responsibilities to machines a map-scheduling set of rules or truly a scheduling algorithm. Most present scheduling algorithms placed awesome effort on increasing statistics locality for overall performance development. Among these scheduling algorithms, the Hadoop Fair Scheduler is the de facto industry fashionable. It deploys a way called delay scheduling, which delays a few map obligations for a small amount of time to gain higher locality. While the information locality problem has received quite a few attention and scheduling algorithms that improve facts locality were proposed in the literature and applied in practice, to the best of our know-how, none of the prevailing works have studied the essential limits of MapReduce computing clusters with information locality. Basic questions along with what is the capacity area of a MapReduce computing cluster with statistics locality, which scheduling set of rules can acquire the total ability

[Veerala, 5(7): July 2018] ISSN 2348 – 8034
DOI: 10 10.5281/zenodo.1310778 Impact Factor- 4.022
region, and how to minimize the waiting time and congestion in a MapReduce computing cluster with statistics locality, continue to be open.

We tend to jointly contemplate data partition and aggregation for a Map-Reduce job with an objective that is to reduce the whole network traffic. Above all, we tend to propose a distributed algorithm for big data applications by moldeling the initial large-scale disadvantage into many sub problems which can be solved in parallel. An online algorithm is developed to perform with the information partition and aggregation throughout a dynamic manner. Finally, simulation results describe that to provide can reduce network traffic worth in every offline and online cases.

Although MapReduce became originally designed as a batch-oriented system, it's far often used for interactive data evaluation: a user submits a job to extract data from a facts set, and then waits to view the outcomes earlier than intending with the subsequent step within the data evaluation technique. This trend has accelerated with the development of highlevel question languages which are finished as MapReduce jobs, which include Hive, Pig, and Sawzall. Traditional MapReduce implementations provide a terrible interface for interactive statistics analysis, because they do now not emit any output till the process has been executed to of completion. However, in lots of instances, an interactive consumer would pick a "quick and grimy" approximation over a accurate answer that takes plenty longer to compute. In the database literature, on-line aggregation has been proposed to address this problem, however the batch-oriented nature of conventional MapReduce implementations makes those strategies difficult to use.

II. RELATED WORK

F. Chen, M. Kodialam, and T. Lakshman studied MapReduce job scheduling with consideration of server project. They confirmed that without such joint consideration, there can be exquisite performance loss. They formulated a MapReduce server-activity organizer hassle. This trouble is NP-complete and we advanced a three- approximation set of rules MarS. They evaluated their set of rules thru enormous simulation. The effects shown that MarS can outperform contemporary strategies by way of as tons as 40% in phrases of total weighted process completion time; they also enforce a prototype of MarS in Hadoop and take a look at it with test on Amazon EC2.

Jianwu Wang et al established one of a kind DDP patterns should have a wonderful impact on the performances of the identical device. They discover that although MapReduce can be used for wider range of applications with both one or two enter datasets, it is not constantly the high-quality preference in phrases of application complexity and overall performance. To recognize the variations, they recognized two affecting elements, namely enter facts balancing and fee sparseness, on their overall performance differences. The feasibility of these factors is verified thru experiments. They consider many gear in bioinformatics and different domains have a comparable logic with CloudBurst as they need to in shape enter datasets, and therefore may also benefit from their findings.

III. FRAME WORK

The system projected a distributed algorithm massive data applications by moldering the initial massive scale problem into several sub issues that may be resolved in parallel The system investigate network traffic reduction among Map reduce jobs by collectively exploiting traffic-aware intermediate data partition and data aggregation among multiple map tasks it is mainly utilized in large scale massive data Applications in that two models are chiefly used like Map phase and reduce introduce that applications based on such models place heavily data-dependency or communication on Virtual Machines; thus network traffic becomes the bottleneck of jobs. The following three phases of information exchange among the execution methodology of an application based on Map reduce model. This paper purpose to the provisioning a virtual cluster in step with the position relationship between Virtual Machines thus on decrease the network traffic and improve the performance of Map phase and Map reduce phase like applications rather than modifying the task scheduling methods or Virtual Machine configurations. By optimizing the design of virtual clusters, cloud users can get a more efficient platform with an equivalent resource request and value, and cloud providers will get an improved resource utilization ratio.

[Veerala, 5(7): July 2018]
DOI: 10 10.5281/zenodo.1310778

ISSN 2348 – 8034
Impact Factor- 4.022

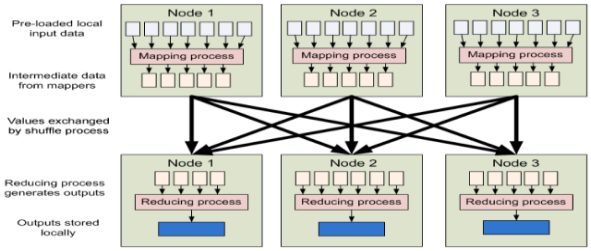


Figure1: Architecture of MapReduce Processing

The most contributions of this paper are summarized below throughout this system to measure the affinity by method the distance of a virtual cluster the shorter the gap, the nearer the virtual cluster. The shortest distance disadvantage is presented to obtain the closest virtual cluster to resolve the shortest distance drawback by formulating it into variety applied math. A heuristic Virtual Machine placement rule is suggests provisioning a virtual cluster. It is designed for Map reduces applications to enhance the shuffle speed and stimulate the execution. It is additionally optimize the virtual cluster from the world, i.e., provisioning virtual clusters for a request queue instead of one request. The online heuristic Virtual Machine placement algorithm and the global optimization algorithm are compared by simulations. The previous has lower time quality whereas the latter arrival shorter average distance for multiple requests to research the performance of our approach through experiments. Within the experiment, describe the support different virtual cluster architectures to check different Map reduce applications. Two metrics of application runtime and cluster compatibility show the potency of virtual cluster optimization. The following figure a pair of shown the simple Map reduce task victimization key with aggregation. The system projected a distributed algorithm for big data applications by moldering the first large scale drawback into many sub problems which is able to be resolved in parallel the system investigate network traffic reduction inside Map reduce jobs by conjointly exploiting traffic-aware intermediate data partition and data aggregation among multiple map tasks. It offers computers as physical or additional usually as virtual machines. A group of virtual machines, virtual bunch is generally requested as a platform for users to run parallel or distributed applications like Map reduce and dryad applications. thus on get high throughput, fast response, load balance, low cost, and low value, several topics on virtual machines configuration, virtual machines placement, virtual machines consolidation, and virtual machines migration are explored.

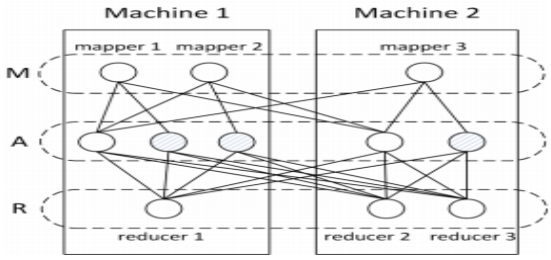


Figure2: Architecture of Three-Layer Model for the Network Traffic Minimization Problem

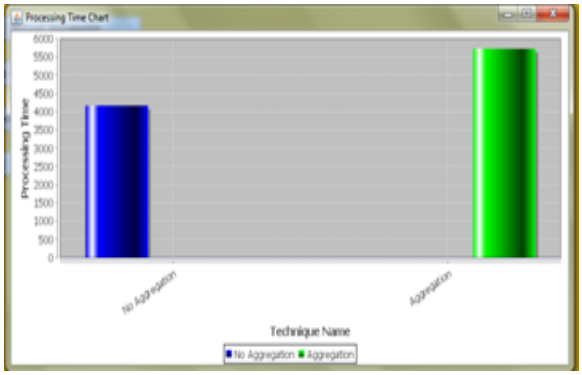
The configuration of a virtual cluster includes a very important impact on the execution of applications running thereon as a result of the physical nodes wherever virtual machines are situated are going to be joined in many ways. For example, some nodes are settled within the same rack whereas others in many racks through a slow link. Map and reduce tasks could part overlap under some cases but the execution is to extend system throughput, and it is troublesome to estimate system parameter set a high accuracy for large information applications. An online algorithm to dynamically adjust data partition and aggregation during the execution of map and reduce tasks is so motivated. The basic plan of this algorithm is to postpone the migration operation till the cumulative traffic cost exceeds a threshold. Another is online algorithm which is additionally designed to deal with the info partition and aggregation in a very dynamic manner. The simulation results finally demonstrated recommend that our proposals will significantly reduce network traffic value in both offline and online cases.

ISSN 2348 – 8034
Impact Factor- 4.022

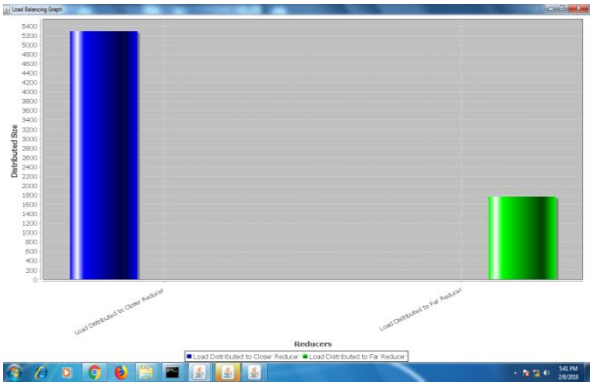
IV. EXPERIMENTAL RESULTS

In our experiments we have described a map reduce framework used for parallel processing large amount of data. Although there are many existing works which have been focused on the traffic reduction, they did not do well in the map reduce paradigm. Those works focused mainly on the map and reduce phases instead of concentrating on shuffle phase. They did not attempt to reduce the data traffic for the improvement of the network traffic and the data cost. Although the existing works focused on the traffic improvement they utilized the large number of keys in the system to make a process more complex. In this paper, we have implemented an efficient system for map reduces jobs by data partition and aggregation for the big data applications.

In the below chart we can observe that difference between the length of both Aggression and No Aggression time.



We can observe that Aggression length is higher than No Aggression length. The difference will be shown in the sense of Processing Time. Through our implementation we have implemented an efficient system for map reduces jobs by data partition and aggregation for the big data applications so we can consider that we are improving the network traffic performance and data cost when compare to existing process.



And we also implemented the load balancing mechanism & in this experiment we defined the several reducers for load balancing. Here, the map can distribute the big data to the multiple reducers. But, which reducers are near to the mapper, those can receive the high data. Which are far to mapper those are can receive less data.

V. CONCLUSION

Finally, we conclude in this paper we proposed a load balancing technique to reduce the burden of the reducer which is receiving data from the mapper. By using this proposed load balancing method, we can improve the performance of MapReduce for big data processing. From the experimental results, we can observe that the network data

[Veerala, **5(7): July 2018]** **ISSN 2348 – 8034**
DOI: 10 10.5281/zenodo.1310778 **Impact Factor- 4.022**
partitioned as well aggregated by using MapReduce and we efficiently minimized the load of the reducer by implementing multiple reducers for one mapper in the network.

REFERENCES

1. *W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "Map task scheduling in mapreduce with data locality: Throughput and heavy-traffic optimality," in INFOCOM, 2013 Proceedings IEEE. IEEE, 2013, pp. 1609–1617*
2. *R. Liao, Y. Zhang, J. Guan, and S. Zhou, "Cloudnmf: A mapreduce implementation of nonnegative matrix factorization for largescale biological datasets," Genomics, proteomics & bioinformatics, vol. 12, no. 1, pp. 48–51, 2014*
3. *G. Mackey, S. Sehrish, J. Bent, J. Lopez, S. Habib, and J. Wang,"Introducing map-reduce to high end computing," in Petascale Data Storage Workshop, 2008. PDSW'08. 3rd. IEEE, 2008, pp. 1–6.*
4. *W. Yu, G. Xu, Z. Chen, and P. Moulema, "A cloud computing based architecture for cyber security situation awareness," in Communications and Network Security (CNS), 2013 IEEE Conference on. IEEE, 2013, pp. 488–492.*
5. *F. Chen, M. Kodialam, and T. Lakshman, "Joint scheduling of processing and shuffle phases in mapreduce systems," in INFOCOM, 2012 Proceedings IEEE. IEEE, 2012, pp. 1143–1151*
6. *Jianwu Wang, Daniel Crawl, Ilkay Altintas, Kostas Tzoumas, Volker Markl, "Comparison of Distributed Data-Parallelization Patterns for Big Data Analysis: A Bioinformatics Case Study", DataCloud'13, Nov. 17, 2013, Denver, CO, U.S.A*
7. *J. Rosen, N. Polyzotis, V. Borkar, Y. Bu, M. J. Carey, M. Weimer, T. Condie, and R. Ramakrishnan, "Iterative mapreduce for large scale machine learning," arXiv preprint arXiv:1303.3517, 2013.*
8. *S. Venkataraman, E. Bodzsar, I. Roy, A. AuYoung, and R. S. Schreiber, "Presto: distributed machine learning and graph processing with sparse matrices," in Proceedings of the 8th ACM European Conference on Computer Systems. ACM, 2013, pp. 197– 210.*
9. *T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, K. Elmeleegy, and R. Sears, "Mapreduce online." in NSDI, vol. 10, no. 4, 2010, p. 20.*
10. *J. Lin and C. Dyer, "Data-intensive text processing with mapreduce," Synthesis Lectures on Human Language Technologies, vol. 3, no. 1, pp. 1–177, 2010*