ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 5, May 2018

Implementing an Effective Online Algorithm for Reducing Network Traffic Cost through the MapReduce Job in Online Manner

[1] Vattipally Latha, [2] Dr. M. Rama Subramanian
[1] M. Tech Student, [2] Professor & Head, [1][2] Department of CSE, Sridevi Women's Engineering College, Village VattiNagulaPally, Mandal Rajendra Nagar, District RangaReddy, Telangana, India

Abstract: **- Most existing work makes of MapReduce efficiency development via optimizing its information transmission. Additionally to information partition, many efforts have been made on nearby aggregation, in-mapper combining and in-network aggregation to decrease network visitors inside MapReduce jobs. The goal of enhancement of network visitors is done with the aid of utilizing partition and aggregation. In line with typical system a hash function is used to partition intermediate knowledge amongst scale back duties however the natural operate will not be efficient to control network traffic. A new intermediate data partition scheme is designed to lessen network user's rate in MapReduce. The aggregator placement concern is regarded, where each and every aggregator can slash merged traffic from more than one map duties. On this paper, we are studying the joint optimization of intermediate data partition and aggregation in MapReduce to lessen network traffic cost for large information purposes. We advocate a three-layer model for this hindrance and formulate it as a mixed-integer nonlinear main issue, which is then transferred into a linear kind that may be solved by way of mathematical tools.**

Key words: - MapReduce, Network Traffic.

I. INTRODUCTION

Big data is an open-source structure that makes it possible for to store & procedure significant data in a dispensed atmosphere across clusters of computer systems utilizing simple programming units. It is designed to scale up from single servers to 1000's of machines, each and every providing local computation and storage. The core of Hadoop consists of a storage part, known as Hadoop Distributed File System (HDFS), and a processing part referred to as Map shrink. Hadoop splits records into giant blocks and distributes them across nodes in a cluster. To procedure knowledge, Hadoop transfers packaged code for nodes to process in parallel situated on the information that wants to be processed. This method takes expertise of data locality nodes manipulating the information they have entry to permit the dataset to be processed faster and more successfully than it could be in a more conventional supercomputer architecture that depends on a parallel file approach where computation and data are allotted via excessive-velocity networking. Big data is for essentially the most section accretion of expertise units so well-known and multifaceted that it's remarkably rough to handle them utilizing close via database admin devices. The principle challenges with giant databases comprise inquiry, production, examination, sharing and perception and stockpiling. As a topic of first significance, understanding is procured from different sources, for illustration, on-line networking, long-established sensor knowledge or project information and so forth. Flume can also be utilized to riskless expertise from online networking. At that factor, this know-how can be gathered utilizing conveyed deed frameworks, for illustration, Google File process. These frameworks are very in a position when number of peruses are high when contrasted with composes.

When working out the efficiency of MapReduce methods; it's easy to view a MapReduce job as which include three phases instead than two phases. The additional section, which is viewed between the map phase and the shrink segment, is an information switch phase referred to as the `shuffle' segment. Within the shuffle phase, the output of the map section is recombined and then transferred to the compute nodes that are scheduled to participate in corresponding slash operations. The performance of MapReduce systems certainly is dependent closely on the scheduling of duties belonging to these three phases even though many efforts were made to beef up the performance of MapReduce

jobs, they exhibit blind observe to the network traffic generated within the shuffle section, which performs a significant role in performance enhancement. In traditional means, a hash perform is used to partition intermediate information among reduce duties, which, nevertheless, isn't visitors-effective on the grounds that we don't recall network topology and knowledge size associated with each key. On this paper, via designing a novel intermediate information partition scheme we slash network site visitor's price for a MapReduce job. Configuring the job, submitting it, controlling its execution, and querying the state are distributed to consumer with the aid of Hadoop. Every job contains impartial tasks, and the entire tasks have got to have a method slot to run. All scheduling and allocation selections in Hadoop are made on a assignment and node slot degree for each the map and reduce phases. The Hadoop scheduling mannequin is a grasp/Slave (master/worker) cluster constitution. The grasp node (JobTracker) coordinates the worker machines (TaskTracker). JobTracker is a method which manages jobs, and TaskTracker is a system which manages tasks on adjoining nodes. The scheduler resides in the Jobtracker and allocates to TaskTracker quite a lot of resources to jogging tasks: Map and scale down tasks are granted impartial slots on each computer.

II. RELATED WORK

The literature survey defines previous working details of some writer concerning equal subject. By deciding on the methodologies and strategies of them we are going to construct an efficient one procedure to retrieve tremendous information.

Chi Yang et all explains a method on A Time effective process for Detecting blunders in colossal Sensor information on Cloud, introduces as colossal sensor knowledge is generic in each industry and scientific research functions the place the information is generated with high volume and speed it's complex to approach making use of on-hand database administration tools or normal data processing applications. Cloud computing supplies a promising platform to support the addressing of this project as it provides a flexible stack of huge computing, storage, and application offerings in a scalable manner at low fee. Some techniques had been developed in up to date years for processing sensor knowledge on cloud, corresponding to sensor-cloud. Nonetheless, these procedures do not provide efficient support on rapid detection and finding of mistakes in enormous sensor knowledge sets. Hadassa Daltrophe, Shlomi Dolev and Zvi Lotker introduces information interpolation situated Aggregation. Given a large set of dimension sensor data, in direct to recognize an easy operate that captures the importance of the information assembled by way of the sensors, we propose for representing the data with (spatial) capabilities, in distinctive with polynomials. Given a (exampled) set of values, we interpolate the datapoints to explain a polynomial that might signify the data. The interpolation is value, given that in undertaking the information is able to be noisy and even Byzantine, the place the Byzantine information stand for an adversarial worth that isn't confined to being close to the proper measured data. The managing of significant knowledge structure also presents interest for the disbursed interpolation process. The thought of gigantic information occurs to some of the essential duties within the occurrence of the massive quantity of information generated by way of nowadays. Speaking and examining the whole data does not extent, even when data aggregation approaches are employed. This recommends a technique to symbolize the distributing huge information by using a handy conceptual perform so that it will direct to efficient utilize of that data. To overcome the above limit, produce two solutions, one that expands the Welch-Berlekamp method within the case of multidimensional knowledge, and copes with discrete noise and Byzantine information, and the subsequent one is established on Arora and Khot methods, increasing them within the case of multidimensional noisy and Byzantine knowledge. For the period of the research we comprise illustrious two different measures for the polynomial becoming to the Byzantine noisy data crisis: the primary being the Welsh-Berlekamp simplification for discrete-noise multidimensional knowledge and the 2d being the linear-programming estimate for multivariate polynomials. Approached by using the error-correcting code systems, we have now encouraged a method to signify a loud malicious input with a multivariate polynomial.

III. FRAMEWORK

In this paper, we tend to together consider knowledge partition and aggregation for a MapReduce job with an objective that's to reduce the whole network traffic. Particularly, we propose a distributed rule for big data applications by molding the initial large-scale drawback into many sub issues which will be resolved in parallel. Moreover, an online algorithm is intended to influence the data partition and aggregation in a very dynamic manner.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 5, May 2018

A. MapReduce Working

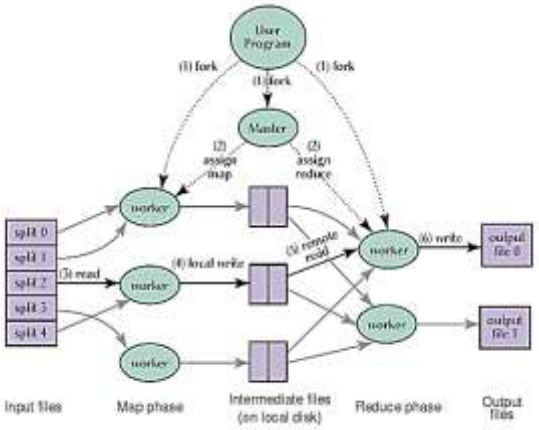


Fig. 1 Execution flow of MapReduce

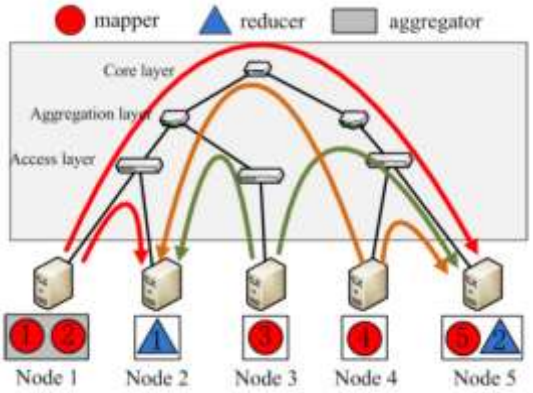
The overall flow of a MapReduce operation which fits through the following sequence of actions:

1. The input file of the MapReduce program is split into M items and starts up several instances of the program on a cluster of machines.
2. One among the instances of the program is elective to be the original whereas the remainders are thought of as workers that are appointed their work by the original. Above all, there are M map tasks and R reduce tasks to assign. The master picks idle workers and assigns each or a lot of map tasks and/or reduces tasks.
3. A worker who is appointed a map task processes the contents of the equivalent input split and generates key/value tries from the input file and passes every pair to the user-defined Map function. The intermediate key/value pairs created by the Map function are buffered in memory.
4. Sometimes, the buffered pairs are written to native disk and partitioned off into R regions by the partitioning function. The locations of those buffered pairs on the native disk ar passed back to the master, who is accountable for forwarding these locations to the scale back workers.
5. Once a reduce employee is notified by the master regarding these locations, it reads the buffered knowledge from the native disks of the map workers that is then sorted by the intermediate keys in order that all occurrences of an equivalent key are classified along. The sorting operation is required as a result of generally many alternative keys map to an equivalent reduce task.
6. The reduce employee passes the key and also the corresponding set of intermediate values to the user's scale back function. The output of the reduce function is added to a final computer file for these reduce partition.
7. Once all map tasks and reduce tasks are completed, the master program wakes up the user program. At this time, the MapReduce invocation within the user program returns the program management back to the user code.

B. System Overview

Our network topology is founded on three tier architectures:

1. Access tier
2. Aggregation tier
3. Core tier



From above figure, the entry tier is made from price-robust Ethernet switches connecting rack VMs. The access switches are connected via Ethernet to a collection of aggregation switches which in flip are linked to a layer of core switches. An inter-rack hyperlink is essentially the most contentious useful resource as all the VMs hosted on rack transfer knowledge throughout the link to the VMs on other racks. Our VMs are allotted in three unique racks, and the map-lower tasks are scheduled as in determine. For example, rack 1 consists of node 1 and 2; mappers 1 and a couple of are scheduled on node 1 and reducer 1 is scheduled on node 2. The intermediate information forwarding between mapper and reducers must be transferred throughout the network. The hop distances between mappers and reducers are proven in determine, e.g., mapper 1 and reducer 2 has a hop distance 6.

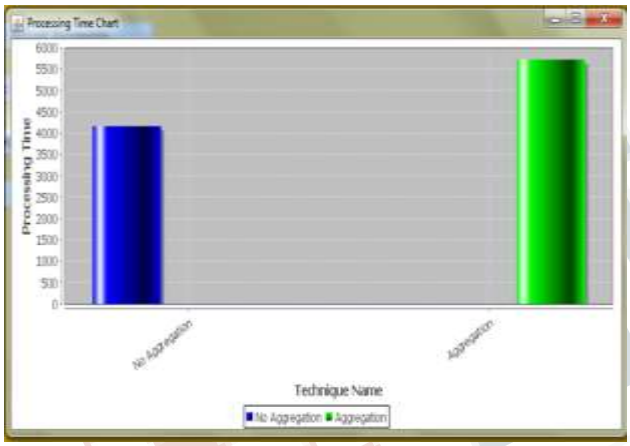
IV. EXPERIMENTAL RESULTS

In this paper, we perform experiments on MapReduce jobs. In this experiment, we run the reducers and define the location values with latitude and longitude. After this, we upload documents as an input to send in the network. After giving input, we have to start the MapReduce aggregation. It will take some time to processing the uploaded data and it displays processing time as well as aggregated data on the screen.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 5, Issue 5, May 2018

Finally, we get the aggregated data and we can get the processing time chart.



Through this experiment we can say our application can significantly reduce the network traffic cost.

V. CONCLUSION

In this paper we can conclude that the main objective of this paper is to reduce the network traffic cost for big data application by using three-tier model. In this proposed model we considered data partition and aggregation techniques by using MapReduce. This MapReduce model works based on two primitives: Map() and Reduce(). Finally, we prove that our proposed system efficiently reduce the network traffic cost.

REFERENCES

[1] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," Communications of the ACM, vol. 51, no. 1, pp. 107–113, 2008.

[2] 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.

[3] 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.

[4] Y. Wang, W. Wang, C. Ma, and D. Meng, "Zput: A speedy data uploading approach for the hadoop distributed file system," in Cluster Computing (CLUSTER), 2013 IEEE International Conference on. IEEE, 2013, pp. 1–5.

[5] T. White, Hadoop: the definitive guide: the definitive guide. " O'Reilly Media, Inc.", 2009.

[6] S. Chen and S. W. Schlosser, "Map-reduce meets wider varieties of applications," Intel Research Pittsburgh, Tech. Rep. IRP-TR-08-05, 2008.

[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] A. Matsunaga, M. Tsugawa, and J. Fortes, "Cloudblast: Combining mapreduce and virtualization on distributed resources for bioinformatics applications," in eScience, 2008. eScience'08. IEEE Fourth International Conference on. IEEE, 2008, pp. 222–229.

[10] J. Wang, D. Crawl, I. Altintas, K. Tzoumas, and V. Markl, "Comparison of distributed data-parallelization patterns for big data analysis: A bioinformatics case study," in Proceedings of the Fourth International Workshop on Data Intensive Computing in the Clouds (DataCloud), 2013.