OPTIMIZATION OF WORK LOAD USING MAP REDUCE FRAMEWORK: Review Study

Aasheesh Raizada[1], Manoj Rana[2] and Pankaj Kumar Varshney[3]
[1]Research Scholar, IET, Mangalayatan University, Aligarh, India
[2]Associate Professor, IET, Mangalayatan University Aligarh, India
[3]Associate Professor, Department of Computer Science, IITM Janakpuri, New Delhi, India
rsgroup.raizada@gmail.com, manoj.rana@mangalayatan.edu.in, pankaj.surir@gmail.com

Abstract

The term Optimize is "to make perfect". It's means choosing the best element from some set of available alternatives. Within the past few years, organizations in diverse industries have adopted Mapreduce framework for large-scale data processing. As we know that Mapreduce has developed to new users for important new workloads have emerged which feature many small, short, and increasingly interactive jobs in addition to the large, long-running batch jobs. In this paper researchers try to focus on optimization of workload in different field such as e-commerce, media and data handling. Mapreduce workloads are driven by interactive analysis, and make heavy use of query like programming frameworks on top of Mapreduce. Mapreduce frameworks can achieve much higher performance by adapting to the characteristics of their workloads.

Keywords: Big Data, Map Reduce, HADOOP, HDFS, YARN, Optimization, Workload.

1. Introduction

It is clear that optimization is the process of modifying a software system to make some aspect of work more efficiently. Optimization will generally focus on improving just one or two aspects of performance, execution time, memory usage, disk space, bandwidth, power consumption or some other resources. Similarly there is no automatic process to reduce or optimize workload but yes there are some tips or we can say that there are some steps to optimize work load using mapreduce such as:

(a) The first step to optimizing Mapreduce performance is to make sure the cluster configuration has been tuned. Mapreduce jobs are fault tolerant, but dying disks can cause performance to degrade as tasks must be re-executed. If we find that a particular task tracker becomes blacklisted on many job invocations, it may have a failing drive.

(b)Tune the number of map and reduce tasks appropriately. Increase the number of mapper tasks to some multiple of the number of mapper slots in the cluster. If we have 100 map slots in the cluster, try to avoid having a job with 101 mappers – the first 100 will finish at the same time, and then the 101st will have to run alone before the reducers can run. This is more important on small clusters and small jobs.

(c) Write a Combiner. We can use a Combiner in order to perform some kind of initial aggregation before the data hits the reducer. The Mapreduce framework runs combiners intelligently in order to reduce the amount of data that has to be written to disk and transferred over the network in between the Map and Reduce stages of computation**.**

Workload optimization allows an application or group of applications to exploit the underlying hardware and infrastructure or middleware layers to achieve maximum performance. The workload is the amount of processing that the computer has been given to do at a given time. The workload consists of some amount of application programming running in the computer and usually some number of users connected to and interacting with the computer's applications. It provides a cost-effective storage solution for large data volumes with no format requirements. At the heart of Hadoop is Mapreduce which is the programming paradigm that allows for scalability. Mapreduce is one of two main components of Hadoop. These are **HDFS** (**H**adoop **D**istributed **F**ile **S**ystem) and **YARN** (**Y**et **A**nother **R**esource **N**egotiator).

There is a growing number of Mapreduce applications such as personalized advertising, sentiment analysis, spam and fraud detection, real time event log analysis, etc.

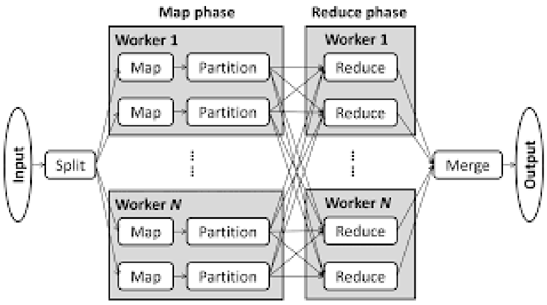

Figure 1: Map-reduce Framework

The workload optimized system is an approach to co-optimize the whole system stack, which includes compilers, language runtimes, and middleware, by exploiting such features for our target workloads ranging from commercial server applications, business analytics, and HPC applications.

The Map Reduce frame work is proposed by Google which provides an efficient and scalable solution for working large-scale data. The basic concept of Map Reduce framework is used to distribute the data among many nodes and process them in parallel manner.

Hadoop is a highly scalable storage platform designed to process very large data sets across hundreds to thousands of computing nodes that operate in parallel. It provides a cost-effective storage solution for large data volumes with no format requirements. At the heart of Hadoop is Mapreduce, the programming paradigm that allows for scalability. Mapreduce is one of two main components of Hadoop. These are HDFS and YARN.

The full form of MAPPER is

M – Maintain

P – Prepare,

P – Produce,

E – Executive,

R – Report.

MAPPER is a processing system which maintain the data and generates a new <key, value> pairs. The <key, value> pairs can be completely different from the input pair the output is the full collection of all these <key, value> pairs. Before writing the output for each Mapper task, partitioning of output take place on the basis of the key and then sorting is done. **Reducer** takes the output.

The **Mapper** (intermediate key-value pair) processes each of them to generate the output. The output of the reducer is the final output.

This image can demonstrate mapreduce easily –

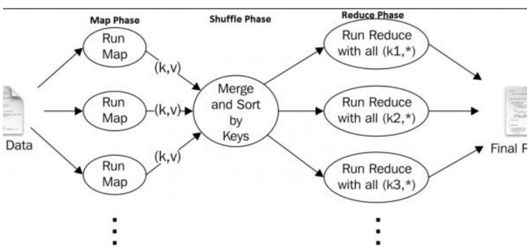

Figure 2: Mapper processing System

- The Map Reduce algorithm contains two important tasks, namely Map and Reduce.
- Mapper takes the input, tokenizes, maps and sorts it. The output of Mapper is used as input by Reducer, which in turn searches matching pairs and reduce.
- The reduce task is always performed after the map job. Map Reduce implements various mathematical algorithms to divide a task into small parts and assign them to multiple systems. In technical terms, Map Reduce algorithm helps in sending the Map & Reduce the network and disks.
- The programming model is simple yet expressive. A large number of tasks can be expressed as Map Reduce jobs. The model is independent of the underlying storage system and is able to process both structured and unstructured data.
- It achieves scalability through block-level scheduling. The runtime system automatically splits the input data into even-sized blocks and dynamically schedules the data blocks to the available nodes for processing.

2. Previous Studies

Zhao S. and Medhi D. [1] focused a Software-Defined Network (SDN) approach in an Application-Aware Network (AAN) platform that provides both underlying networks functions as well Map Reduce particular forwarding logics.

Gopal V. K. and Jackleen I. K. [2] proposed the idea of estimation of the values of parameters: filter ratio, cost of processing a unit data in map task, cost of processing a unit data in reduce task, communication cost of transferring unit data. The result shows that for a particular data size with increasing deadlines, resource demand will decrease. If the data size increases and deadline is kept constant, resource demand will increase. Thant P. T., Powell C. and Sugiki A. [3] addressed problems by optimizing the instance resource usage and execution time of Map Reduce tasks using a multi objective steady-state Non-dominated Sorting Genetic Algorithm II (SSNSGA-II) approach. The instance resource usage cost of Map Reduce tasks is calculated based on the cost of machine instance types and the number of machine instances in the Hadoop cluster. The optimized configuration is identified by selecting an optimal setting that satisfies two objective functions associated with instance resource usage and execution time minimization, from Pareto optimal front solutions. Although dynamic machine instance type is considered within the search process in our system, dynamic cluster size is out of consideration and intended to be carried out in our future. Experiments conducting using workloads from the HiBench benchmark on a high specification 6-node Hadoop cluster verify the efficacy of our proposed approach.

Xu H. and Lau W. C. [4] focused on the design of speculative execution schemes for parallel processing clusters from an optimization perspective under different loading conditions. For the lightly loaded case, analyze and propose one cloning scheme, namely, the Smart Cloning Algorithm (SCA) which is based on maximizing the overall system utility. The workload there holds under which SCA should be used for speculative execution. For the heavily loaded case, the Enhanced Speculative Execution (ESE) algorithm which is an extension of the Microsoft Mantri scheme to mitigate stragglers. The simulation results showed SCA reduces the total job flow time, i.e., the job delay/ response time by nearly 6% comparing to the speculative execution strategy of Microsoft Mantri. The ESE Algorithm outperforms the Mantri baseline scheme by 71% in terms of the job flow time while consuming the same amount of computation resource.

XU H. and LAU W. C. [5] showed SCA can reduce the total job flow time by nearly 22% comparing to the speculative execution strategy of Microsoft Mantri. Execution (ESE) algorithm an extension of the Microsoft Mantri scheme show that the ESE algorithm can beat the Mantri baseline scheme by 35% in terms of job flow time while consuming the same amount of resource.

Kim Ye S. et. al. [6] presented a modified Map Reduce approach focused on the iterative clustering algorithms in the Apache Mahout machine learning library that leverage the acceleration potential of the Intel integrated GPU in a multi-node cluster environment. The accelerated framework shows varying levels of speed-up (≈45x for Map tasks-only, ≈4.37x for the entire K-means clustering) as evaluated using the HiBench benchmark suite. Various experiments and in-depth analysis, find that utilizing the integrated GPU via Open CL offers significant performance and power efficiency gains over the original CPU based approach.

Zhang Z., Cherkasova L. and Loo B. T. [7] demonstrate that the application performance of a customer workload may vary significantly on different platforms. This makes a selection of the best cost/performance platform for a given workload being a challenging problem. Evaluation study and experiments with Amazon EC2 platform reveal that for different workload mixes the optimized platform choice may result in 37-70% cost savings for achieving the same performance objectives when using different choices. The results of this simulation study are validated through experiments with Hadoop clusters deployed on different Amazon EC2 instances.

Sivaranjani V. and Jayamala R. [8] proposed the workload of jobs in clusters mode using Hadoop. Map Reduce is a programming model in Hadoop used for maintaining the workload of the jobs. Depend on the job analysis statistics the future workload of the cluster is predicted for potential performance optimization by using genetic algorithm.

Ding D., Dong F. and Luo J. [9] described a Multi-queries optimization framework based on Map Reduce-oriented cloud environment (Multi-Q), which utilizes the dependence between multiple queries to realize query results reuse. Firstly, a cluster-based partition algorithm called CPA has been exploited to conduct the logic partition of the search range of query workload. Secondly, a multi-queries reuse dependence graph (MRDG) construction method on the basis of the cluster-based partition results has been presented to depict the dependence between the multiple queries. Finally, a Multi-Q processing algorithm based on Multi-Q Reuse Dependence Graph has been put forward to achieve the query results reuse and improve the overall query processing performance. After evaluating this approach by deploying Multi-Q based on Hadoop in a real cloud environment, called SEU-Cloud, and conducting extensive experiments based on the standard TPC-H. The result verifies that compared with Hive, the performance of improvement is approximately 39.3% by using Multi-Q. Tang S., Lee B. S., and He B. [10] proposed an alternative technique called Dynamic Hadoop Slot Allocation by keeping the slot-based model. It relaxes the slot allocation constraint to allow slots to be reallocated to either map or reduce tasks depending on their needs. Second, the speculative execution can tackle the straggler problem, which has shown to improve the performance for a single job but at the expense of the cluster efficiency. Study also showed Speculative Execution Performance Balancing to balance the performance trade off between a single job and a batch of jobs. Third, delay scheduling has shown to improve the data locality but at the cost of fairness. Another technique called Slot Pre Scheduling which can improve the data locality but with no impact on fairness. Finally, by combining these techniques together, form a step-by-step slot allocation system called Dynamic MR that can improve the performance of Map Reduce workloads substantially. The experimental results show that Dynamic MR can improve the performance of Hadoop MRv1 significantly while maintaining the fairness, by up to 46 _ 115 percent for single jobs and 49 _ 112 percent for multiple jobs.

Hashem I. A. T. et. al. [11] focuses on the relationship between big data and cloud computing, big data storage systems, and Hadoop technology. The study also described those challenges which are investigated, with focus on scalability, availability, data integrity, data transformation, data quality, data heterogeneity, privacy, legal and regulatory issues, and governance. The study focus on classification for big data, a conceptual view of big data, and a cloud services model and then compared with several representative big data cloud platforms. The background of Hadoop technology and its core components which are Map Reduce and HDFS. The

study also presented available Map Reduce projects and related software. This paper covered volume, scalability, availability, data integrity, data protection, data transformation, data quality, privacy and legal or regulatory issues, data access, and governance.
Gunarathne T. et. Al. [12] presented the Twister4Azure iterative Map Reduce runtime and a study of four real world data-intensive scientific applications implemented using Twister4Azure – two iterative applications, Multi-Dimensional Scaling and K Means Clustering; and two pleasingly parallel applications, BLAST + sequence searching and Smith Waterman sequence alignment. Performance measurements show comparable or a factor of 2 to 4 better results than the traditional Map Reduce runtimes deployed on up to 256 instances and for jobs with tens of thousands of tasks. They also focus to present solutions to several factors that affect the performance of iterative Map Reduce applications on Windows Azure Cloud.
Li J. et. al. [13] proposed two online dynamic resource allocation algorithms for the IaaS cloud system with preemptable tasks. The algorithms adjust the resource allocation dynamically based on the updated information of the actual task executions. And the experimental results show that the algorithms can significantly improve the performance in the situation where resource contention is fierce.
Triguero I. et. al. [14] developed a Map Reduce-based framework to distribute the functioning of these algorithms through a cluster of computing elements, proposing several algorithmic strategies to integrate multiple partial solutions (reduced sets of prototypes) into a single one. The paper also proposed model enables prototype reduction algorithms to be applied over big data classification problems without significant accuracy loss.
Hsua H. et. al. [15] proposed a method to improve Map Reduce execution in heterogeneous environments. This is done by dynamically partitioning data before the Map phase and by using virtual machine mapping in the Reduce phase in order to maximize resource utilization. Simulation and experimental results showed an improvement in Map Reduce performance, including data locality and total completion time with different optimization approaches.
Wang L. et. al. [16] presented the design and implementation of G-Hadoop, a Map Reduce framework that aims to enable large scale distributed computing across multiple clusters.
Maheshwari N. Nanduri R. and Varma V. [17] addressed energy conservation for clusters of nodes that run Map Reduce jobs. The algorithm dynamically reconfigures the cluster based on the current workload and turns cluster nodes on or off when the average cluster utilization rises above or falls below administrator specified thresholds, respectively. This study also evaluates our algorithm using the Grid Sim toolkit and show that the proposed algorithm achieves an energy reduction of 33% under average workloads and up to 54% under low workloads.

3. Research Gaps

After review of literature, the researcher has figure out the following gaps:

- Dynamic slot configuration can be used while processing a large data set with Map Reduce paradigm. It can help to optimize the performance of Map Reduce framework, Gopal V. K. and Jackleen I. K. (2017).
- Dynamic cluster resizing and parameter configuration optimization can be used because they speed up Map Reduce processing in Hadoop Hadoop Map Reduce parameter configuration has a significant impact on application performance, Thant P. T., Powell C. and Sugiki A. (2016).
- **R**esearchers can be designed new frameworks and benchmarks, such as iterative and streaming workloads to get new resource results and to adapt current Map Reduce frameworks to support additional resources that can accelerate the execution of the workloads, Veiga J. et. al. (2015).

4. Conclusion

Researchers have discussed scheduling of Map Reduce parallel applications to optimization of workload in different field like e-commerce, social media, communication and etc. It is very difficult to identify the parameters that significantly affect the performance of a particular application that's why parameter configuration optimization to speed up. Mapreduce processing in Hadoop is a daunting and time consuming task. With the help of SHadoop users can shorten the start up and cleanup time of all the jobs which is especially effective for the jobs with short running time. It can benefit most short jobs with large deployment or many tasks.
From the past few years organisations in diverse industry have adopted Mapreduce framework for large scale data processing. Along with new users important new workloads have emerged which feature many small, short and increasingly interactive jobs in addition to the large and long running batch jobs for which Mapreduce frameworks were originally designed .It is important to work with an empirical analysis Mapreduce traces from six separate business critical deployments inside Facebook and Cloudera costumers in telecommunications, e-commerce and retail as well as in media.

Reference

[1] Zhao S. and Medhi D. (2017) "Application-Aware Network Design for Hadoop Map Reduce Optimization Using Software Defined Networking", IEEE, pp. 1-14, DOI 10.1109/TNSM.2017.2728519.

[2] Gopal K. V. and Jackleen K. (2017) "Dynamic Map Reduce for Job Workloads through Slot Configuration Technique", International Journal of Innovative Research in Computer and Communication Engineering, pp. 2199-2203, DOI: 10.15680/IJIRCCE.2017. 0502207.

[3] Thant P. T., Powell C. and Sugiki A. (2016) "Multi Objective Hadoop Configuration Optimization using Steady State NSGA-II", IEEE, pp. 293-298, DOI 10.1109/SCIS&ISIS.2016.160.

[4] Xu H. and Lau W. C. (2016) "Optimization for Speculative Execution in Big Data Processing Clusters", IEEE, pp. 1-17, DOI 10.1109/TPDS.2016.2564962.

[5] Xu H. and Lau W. C. (2015) "Optimization for Speculative Execution in a Map Reduce-like Cluster", IEEE, pp. 1071-1079.

[6] Kim s. y., Bottleson J., Jin J., Bindu P., Sakhare S. C. and Spisak J. S. (2015) "Power Efficient Map Reduce Workload Acceleration using Integrated-GPU", IEEE, pp. 162-169, DOI 10.1109/BigDataService.2015.12.

[7] Zhang Z., Cherkasova L. and Loo B. T. (2014) "Optimizing Cost and Performance Trade-Offs for Map Reduce Job Processing in the Cloud", IEEE, pp. 1-8.

[8] Sivaranjani V. and Jayamala R. (2014) "OPTIMIZATION OF WORKLOAD PREDICTION BASED ON MAP REDUCE FRAME WORK IN A CLOUD SYSTEM", International Journal of Research in Engineering and Technology, Vol. 3, Issue. 3, pp. 264-266.

[9] Ding D., Dong F. and Luo J. (2014) "Multi-Q: Multiple Queries Optimization based on Map Reduce in Cloud", IEEE, pp. 100-107, DOI:10.1109/CBD.2014.20.

[10]Tang S., Lee B-S. and He B. (2014) "Dynamic MR: A Dynamic Slot Allocation Optimization Framework for Map Reduce Clusters", IEEE, Vol. 2, No.3, pp. 333-347, DOI:10.1109/TCC.2014.2329299.

[11]Hashem I. A. T. Yaqoob I. Anuar N. B. Mokhtar S. Gani A. and Khan S. U. (2013) “The rise of “big data” on cloud computing: Review and open research issues”, ELSEVIER, pp. 98-115, DOI 10.1016/j.is.2014.07.006.

[12]Gunarathne T. Zhang B. Wu T. L. and Qiu J. (2013) “Scalable parallel computing on clouds using Twister4Azure iterative Map Reduce”, ELSEVIER, pp. 1035-1048, DOI 10.1016/j.future.2012.05.027.

[13]Li J., Qiu M., Mingb Z., Quanc G., Qin X., and Gue Z., (2012) “Online optimization for scheduling preemptable tasks on IaaS cloud systems”, ELSEVIER pp. 666-676, DOI 10.1016.

[14]Triguero I. Peralta D. Bacardit J. Garcıac S. and Herreraa F., (2015) “MRPR: A Map Reduce solution for prototype reduction in big data classification”, ELSEVIER, pp. 1-37, DOI 10.1016/j.neucom.2014.04.078.

[15]Hsua H. C. Slagter D. K. and Chung C. Y.(2015)”Locality and loading aware virtual machine mapping techniques for optimizing communications in Map Reduce applications”, ELSEVIER, pp. 43-54, DOI 10.1016/j.future.2015.04.006.

[16]Wang L. Taoc J. Ranjand R. Martenc H. Streit A. Chene J. and Chena D.,(2013) “G-Hadoop: Map Reduce across distributed data centers for data-intensive computing”, ELSEVIER, pp. 739-750, DOI 10.1016/j.future.2012.09.001.

[17]Maheshwari N. Nanduri R. and Varma V. (2012) “Dynamic energy efficient data placement and cluster reconfiguration algorithm for Map Reduce framework”, ELSEVIER, pp. 119-127, DOI 10.1016/j.future.2011.07.001.