Map Reduce Performance Evaluation: A Step Towards Big Data Optimization

Shabir Ahmad [1], Shafiq Hussain[2*],Zainab Safdar[2], Mohib Ullah Khan[3], Muhammad Amin Abid[4], Hadiz Tariq Masood[5], Tayyab Ali Raza[6]

Abstract: **There is a remarkable growth in the volume of data processed every day. The revolution has taken us to the level where we are trying to find the technology suitable for the amount of data in pet bytes and zeta bytes. With the rapid change in the market, it has become a compulsion for the business to compete strategically. Decision support plays a vigorous part in this regard. The concern is that there is a dire need for sound technology to process an enormous amount of data and do it with efficiency and reliability. Performance optimization of MapReduce is critical because it leads us to the optimization of big data. Increasing the performance of MapReduce will lead us to an optimized state of big data. In this paper, we explore the performance problems associated with MapReduce's processing and review different models relating to the performance evaluation of MapReduce processing**.

Keywords**: Big Data, MapReduce, Assessment Model, KOALA, Cross-Layer.**

INTRODUCTION

MapReduce is a distributed big data processing framework used by Google. It is based on functional programming, including a Map of function and a reduced function. MapReduce is a strong and fault-tolerant framework used by Google to process tremendous amounts of data [1]. There is a dire need to consider the critical points and issues regarding big data processing by MapReduce. This research will highlight the issues then move towards finding a solution. This research provides a review of work done on the performance evaluation of MapReduce and also discusses the matrix that can be used to increase the performance of MapReduce [2].

Performance of the hardware layer (processors, storage, networks) and the (operating system, JVM runtime, applications) software stack are affected. Each layer has several parameters that could lead to significant performance changes [3]. MapReduce framework consists of several distributed and processed clusters in a distributive environment, as shown in figure 1. The file system divides a Map task into large pieces; each of these map tasks is given a key/value succession, the key/value pairs are written on the local disks and are broken down to N number of reduced tasks. Each of the reduced tasks is assigned to each cluster distributed over the network. Each reduced task works on a single key or the value pair. After the processing is done, the reduced tasks are combined again to show the results [4].

Guanying Wang, Ali R. Butt, Prashant Pandey, and Karan Gupta worked on the MapReduce simulator to evaluate the performance of applications they called this MRPerf [5]. They performed calculations on medium-scale clusters and found out that the simulator was accurately predicting the performance of applications. Experiments revealed that MRPerf resulted in 28.5% increased performance in a cluster that was running Hadoop [6].

Colby Ranger et al. and Songchang [7, 8] worked on the evaluation of MapReduce for multicore and multiprocessor systems. They worked on phonix, which used shared memory to reduce task initiation and communication expenses. It demonstrated that phonix resulted in scalable performance for multicore and multiprocessor systems.

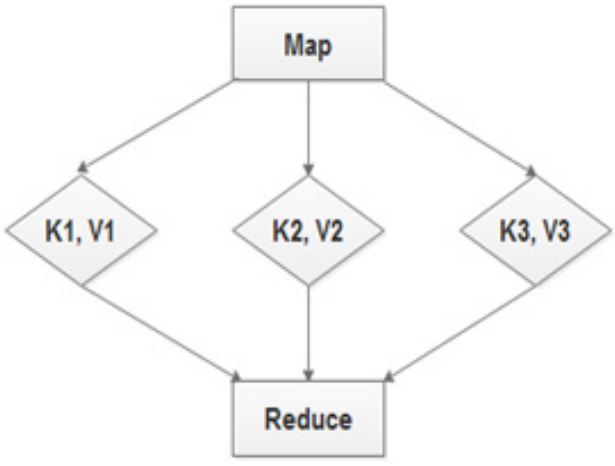

Figure 1. MapReduce Working

Jiong Xie et al. and Novian et al. [9, 10] worked on refining the performance of MapReduce through the placement of Data in heterogeneous clusters. The work was based on placing data across different nodes such that respectively node equally shares a load of data processed. This strategy always improves the performance of MapReduce.

[1] Department of Computer Science, Government College of Commerce, Multan, Pakistan
[2]Department of Computer Science, University of Sahiwal, Sahiwal, Pakistan
[3]Department of Information Technology, Bahauddin Zakariya University, Multan, Pakistan
[4]Department of Chemistry, University of Sahiwal, Sahiwal, Pakistan
[5]Department of Physics, University of Sahiwal, Sahiwal, Pakistan
[6]Department of Computer Science, Barani Institute of Sciences ARID University Rawalpindi, Pakistan
Email: drshafiq@uosahiwal.edu.pk

Xiao Yanga and Jianling Sun worked on the analytical performance model of MapReduce [11]. They proposed a general performance model for MapReduce. Their work suggested that the performance of MapReduce can be significantly enhanced by changing the split granularity and the number of Reducers. Their work provided better knowledge about MapReduce and its performance parameters.

PERFORMANCE MODELS OF MAPREDUCE

Two performance models of MapReduce are used for the optimization of performance.

A) Cross-Layer Performance Assessment Model

Performance evaluation using the cross-layer model has three basic patterns. The framework's core is to associate application actions corresponding to resource consumption [14].

1)Memory Efficiency

This pattern has a pair of assessment indexes called <Split Records, M Memory >. Split Records is the index that defines the number of records, and M Memory is the index that defines the memory that is utilized. Fault tolerance is a major thought while designing Hadoop. In order to avoid the successful jobs carried out multiple times, every output of the Map job is written on the disk. The reduction of the Map jobs will include getting the intermediate data from respective mappers. This phase involves splitting and merging [15]. Figure 2 shows the memory efficiency of the cross-layer performance assessment model.

Split and merge engage I/O operations which take much time. This can cause a bottleneck in the performance of MapReduce. So, the numbers of records split to determine the performance of MapReduce to a great extent. This is why Split Records is chosen as an application index to measure the split efficiency [16].

Split Records also elaborates the number of merges that occur. The optimizing goal related to the map phase is to minimize the number of splits, which means that the split occurs only once. The goal of the optimizing phase differs from the first one. We ideally want Split Records to be equal to zero. Suppose the Split Records is more than the number of reduced records; it shows that the job will be consuming several merges causing high expenses [17].

First, examine whether a value separates memory size from the right to exit and then ensure that the partition was placed properly over the buffer parameter.

2)Schedule Efficiency

This pattern involves <S Shuffle, N network>. The application index S Shuffle signifies the period between the finish of the Map phase and the start of the last reduce phase [13]. N network index indicates network activity while the S Shuffle phase is not recommended. Our opinion pattern data memory for use in a manual separation stage effectively.

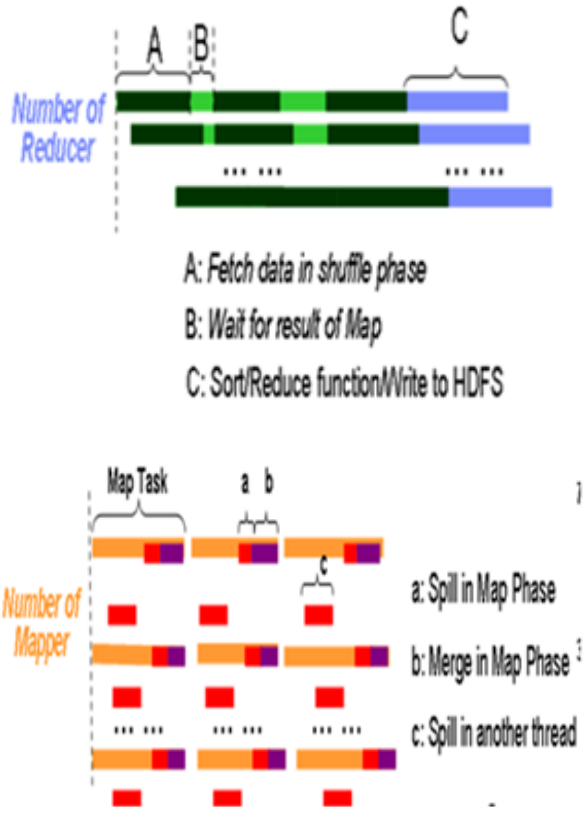

Figure 2. Memory Efficiency of Cross-Layer Performance Assessment Model [12]

The random <S template, N Red> contains. The directory of the random application S represents the time interval between the start of the final reduction phase and the final phase of the map [18]. The pattern works collaboratively to show how good MapReduce programs respond while obtaining intermediate data.

The main feature of MapReduce is that the shuffle stage starts as soon as the initial map finishes. Figure 2 shows by what means the schedule completes. The shuffling phase includes two parts: the first part includes copying intermediate data from mappers. Based on this, we describe S Shuffle; this shows the time it takes for copying in-between data as the Map phase split ends.

The act will be automatically optimized when we have reduced S Shuffle. The reduced the SShuffle is, the better we will achieve the performance. When the amount of data is large, then the throughput is critical. It can cause a bottleneck if the amount of data is large. The pattern completely views both the issues to recognize real performance matter [19, 20].

Table I. Cross-Layer Performance

Pattern	Explanation	Inefficiency Cause
Split Efficiency $R_{Split\ Records}$, M> < $_{Memory}$	$R_{Split\ Records}$: Spill Record number in a Map/Reduce phase M_{Memory}: system memory or JVM heap utilization	Runtime buffer and JVM related parameters
Schedule Efficiency <$S_{Shuffle}$, $N_{network}$>	$S_{Shuffle}$: Map phase End Timing For last reduce (in Wave first) and starts sort $N_{network}$: network utilization	Runtime scheduling parameters
Function Efficiency $F_{supporting\ function}$, > < $C_{CPU/disk}$	$F_{supporting\ function}$: cost of support functions $C_{CPU/disk}$: utilization of CPU and disk I/O	Application Algorithm

3)Functions Efficiency

This group defines a pair of assessment indexes. <Supporting function, CCPU/disk>. In this pattern, the Supporting function indicates the cost of supporting functions and, CCPU/disk shows the consumption of CPU disk. Figure 3 shows an instance of this design.

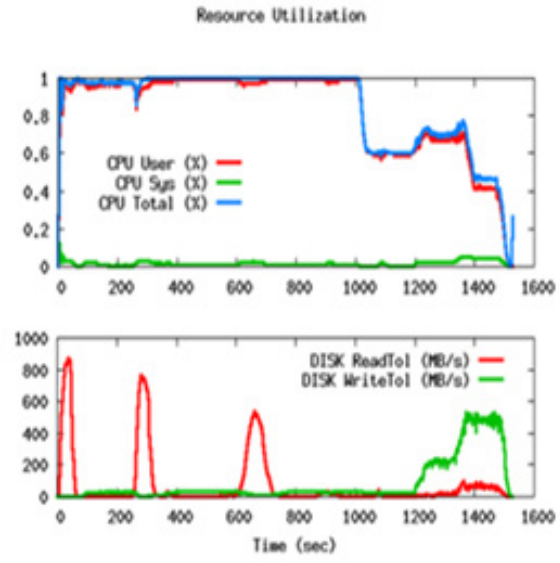

Figure 3. Function Efficiency of Cross-Layer Performance Model [22]

The functions provided by Hadoop are called supporting functions. To achieve optimum performance, we need to pay attention to the number of CPU cycles that the supporting functions consume [21].

B) KOALA Based Big Data Processing System (Performance Assessment Model)

KOALA has a scheduler at the core, taking jobs from the users and assigning them to suitable clusters choosing the cluster based on its scheduling strategies. Jobs are given to KOALA using runners. KOALA uses network information services to evaluate the status of resources. Figure 4 displays the Koala base big data systems.

1) MapReduce Runner

We implement a resource organizing system that makes remote deployment of dynamic MapReduce clusters possible. A MapReduce cluster is based on an implementation framework to help an execution platform for MapReduce applications and a storage layer that helps manage and store a large volume of data.

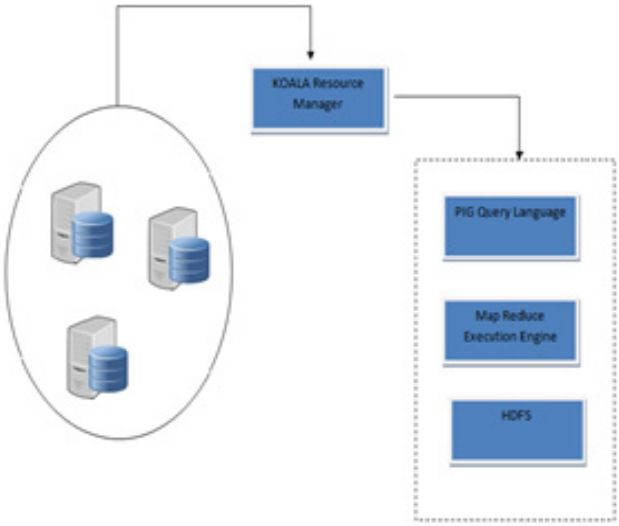

Figure 4. Koala base Big Data System

An MR-runner can set up multi MR clusters inside a solitary physical cluster satisfying all the isolation properties. It can also extend a single MR cluster to a multi-cluster system. If we deploy multiple MR clusters, we achieve the following isolation properties concerning performance, data management, fault acceptance, and version.

Storage layers also need to be regulated to yield optimum benefit to the nodes when the MR group is built [23, 24]. So, we can add several core nodes into an MR cluster; the problem comes when we are going to remove nodes, the problems may include integrity and data availability. We are improving the resize strategy to facilitate expansion and contract operations at the storage layer [25, 26].

2)The Performance Model of KOALA Based Big Data Processing System

The performance model built for KOALA based big data .processing system consists of the following components

Benchmarking

To create a benchmark for the performance assessment, we have focused on two criteria. One of them is the amount of processed data, including multiple types of data, data from academia, social sites, scientific research, and industry. The second is the extent to which the system is producing an accurate result. This will lead us to check the reliability of the system.

Real-World Applications

MapReduce applications are the most common apply algorithms for processing text, web, and machine learning. HiBench [27, 28] is a MapReduce Benchmarking tool that adds such workloads with arbitrarily generated datasets.

3)MapReduce Modeling

Building a performing analysis for the performance of MapReduce, choose a multilevel approach that involves the following.

Infrastructure

MR-Runner enables MapReduce clusters in a multi-cluster environment. There can be three scenarios relating to the deployment of the MapReduce cluster. A single MR cluster can be deployed over a dedicated cluster. A single MR cluster can be deployed over multiple clusters, and multiple MR clusters can be deployed as sharing a single cluster.

Middleware

Middleware includes a set of parameters: memory, storage capacity, and network. With all these parameters, we also try to find the exact configuration of the system, provided with absolute infrastructure. This can lead to a trustworthy performance model of MapReduce.

Application

To build the performance model for MapReduce, we need to understand the extent to which it is suitable a certain amount of workloads. The diversity of workloads can be a strength and a weakness of the system.

COMPARISON OF PERFORMANCE MODELS

We have so far reviewed two models for evaluating the performance of MapReduce. Table 2 shows a comparison of performance evaluation models that we discussed earlier.

Table II. Comparison of Performance Evaluation Models

Performance Evaluation Model	Matrix Involved	Detail Functions
Cross-Layer Performance Evaluation	Memory Efficiency Schedule Efficiency Functions Efficiency	Focuses on three criteria's, making efficient use of memory, scheduling map and reducing ,tasks as good as possible optimizing supporting functions to achieve higher level perfor- .mance
KOALA Based Big Data System Performance Evaluation	Benchmarking MapReduce Modeling Infrastructure Middleware Application	Considering the amount of data processed, the types of data sets, and the accuracy of re- .sults MapReduce Modeling includes infrastructure, middleware, and application criteria to achieve .optimum performance

CONCLUSION

To deal with a large amount of data, we need systems capable enough and reliable in processing large data sets. MapReduce is one of the reliable and fault-tolerant systems in this regard. In this paper, we explored the performance problems connected to the processing of MapReduce and reviewed different models relating to the performance evaluation of MapReduce processing. We trust that the research paper will open ways for others to research more in finding more concrete models for optimizing big data processing and eradicating performance concerns of MapReduce.

FUTURE WORK

This paper discussed how MapReduce is reliable for processing big data. The tests and features of MapReduce and reviewed the performance models for optimizing Big Data processing by MapReduce. Several factors we discussed can lead us to the optimized performance of MapReduce. There are many other issues with the performance of MapReduce and yet many to discover. The research opens a broader gateway to search out an optimized solution that will eliminate the performance issues of MapReduce and ultimately lead us to big data optimization.

REFERENCES

[1] Yang, Christopher, et al. "Osprey: Implementing MapReduce-style fault tolerance in a shared-nothing distributed database." Data Engineering (ICDE), 2010 IEEE 26th International Conference on. IEEE, 2010.

[2] Yan Li , Kun Wang , Qi Guo, Xin Li, Xiaochen Zhang, Guancheng Chen, Tao Liu, Jian Li, IBM Research - China, fliyancrl, wangkun, qiguo, xinlibj, chengc Breakingthe Boundary for Whole-System Performance Optimization of Big Data,

[3] Guanying Wang, Ali R. Butt, Prashant Pandey, Karan

Gupta, Using Realistic Simulation for Performance Analysis of MapReduce Setups

[4] Chen, Yanpei, Sara Alspaugh, and Randy Katz. "Interactive analytical processing in big data systems: A cross-industry study of mapreduce workloads." Proceedings of the VLDB Endowment 5.12 (2012): 1802-1813.

[5] Jeffrey Dean and Sanjay Ghemawat, MapReduce: Simplified Data Processing on Large Clusters COMMUNICATIONS OF THE ACMJanuary 2008/ Vol. 51, No. 1

[6] Aji, Ablimit, et al. "Hadoop gis: a high performance spatial data warehousing system over mapreduce." Proceedings of the VLDB Endowment 6.11 (2013): 1009-1020

[7] Colby Ranger, Ramanan Raghuraman, Arun Penmetsa, Gary Bradski, Christos Kozyrakis, Evaluating MapReduce for Multi-core and Multiprocessor Systems, 1-4244-0805-9/07/$25.00 ©2007 IEEE

[8] Songchang Jin, Shuqiang Yang, Yan Jia, Optimization of Task Assignment Strategy for Map-Reduce International Conference on Computer Science and Network Technology

[9] Jiong Xie, Shu Yin, Xiaojun Ruan, Zhiyang Ding, Yun Tian, James Majors, Adam Manzanares, and Xiao Qin, Improving MapReduce Performance through Data Placement in Heterogeneous Hadoop Clusters 978-1-4244-6534-7/10/$26.00 ©2010 IEEE

[10] Novia Nurain Hasan Sarwar, Md.Pervez Sajjad, Moin Mostakim An In-depth Study of Map Reduce in Cloud Environment 2012 International Conference on Advanced Computer Science Applications and Technologies

[11] Xiao Yang, Jianling Sun, AN ANALYTICAL PERFORMANCE MODEL OF MAPREDUCE, 2011IEEE.

[12] Alexandrov, Alexander, et al. "The stratosphere platform for big data analytics." The VLDB Journal 23.6 (2014): 939-964.

[13] Dittrich, Jens, and Jorge-Arnulfo Quiané-Ruiz. "Efficient big data processing in Hadoop MapReduce." Proceedings of the VLDB Endowment 5.12 (2012): 2014-2015.

[14] Shafer, Jeffrey, Scott Rixner, and Alan L. Cox. "The hadoop distributed filesystem: Balancing portability and performance." Performance Analysis of Systems & Software (ISPASS), 2010 IEEE International Symposium on. IEEE, 2010.

[15] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." Communications of the ACM 51.1 (2008): 107-113.

[16] Zhao, Jiacheng, et al. "An empirical model for predicting cross-core performance interference on multicore processors." Parallel Architectures and Compilation Techniques (PACT), 2013 22nd International Conference on. IEEE, 2013.

[17] Chu, Cheng-Tao, et al. "Map-reduce for machine learning on multicore." Advances in neural information processing systems. 2007.

[18] Katal, Avita, Mohammad Wazid, and R. H. Goudar. "Big data: issues, challenges, tools and good practices." Contemporary Computing (IC3), 2013 Sixth International Conference on. IEEE, 2013.

[19] Zaharia, Matei, et al. "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing." Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. USENIX Association, 2012.

[20] Kalia, K., & Gupta, N. (2021). Analysis of hadoop MapReduce scheduling in heterogeneous environment. Ain Shams Engineering Journal, 12(1), 1101-1110.

[21] Hashem, Ibrahim Abaker Targio, et al. "The rise of "big data" on cloud computing: Review and open research issues." Information Systems 47 (2015): 98-115.

[22] Li, Xiaobing, et al. "Coomr: Cross-task coordination for efficient data management in mapreduce programs." Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. ACM, 2013.

[23] Bogdan Ghit¸ , Alexandru Iosup, Dick Epema, Towards an Optimized Big Data Processing System Bogdan Ghit¸ Delft University of Technology 2013 13th IEEE/ ACM International Symposium on Cluster, Cloud, and Grid Computing pg 83-86

[24] Fan Zhang, Qutaibah. M. Malluhi, Tamer M. Elsyed ConMR: Concurrent MapReduce Programming Model for Large Scale Shared-Data Applications 2013 42nd International Conference on Parallel Processing.

[25] Kumar, D., & Jha, V. K. (2021). An improved query optimization process in big data using ACO-GA algorithm and HDFS map reduce technique. Distributed and Parallel Databases, 39(1), 79-96.

[26] Xu, H., Wang, L., & Zhang, Y. (2022). Analysis of Big Data Processing Based on Mapreduce and Convolution Neural Network. In Innovative Computing (pp. 1253-1257). Springer, Singapore

[27] Chamikara Jayalath, Julian Stephen, Patrick Eugster From the Cloud to the Atmosphere: Running MapReduce across Data Centers IEEE TRANSACTIONS ON COMPUTERS, VOL. 63, NO. 1, JANUARY 2014.

[28] S. Huang, J. Huang, J. Dai, T. Xie, and B. Huang, "The Hibench Benchmark Suite: Characterization of theMapReduce-based Data Analysis," 26th Int'l Conf. on Data Engineering Workshops (ICDEW), pp. 41–51, 2010.