IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 7, Issue 12, December 2018

Big Data Load Management: A Survey

Aasheesh Raizada[1] **Manoj Rana**[2], **Pankaj Kumar Varshney**[3]

Research Scholar, Department of Computer Application, IET, Mangalayatan University, Aligarh, India[1]

Associate Professor, Department of Computer Application, IET, Mangalayatan University, Aligarh, India[2]

Associate Professor, Department of Computer Science, IITM, GGSIP University, New Delhi, India[3]

Abstract: Big Data refers to huge volume of data which present everywhere, in human body as human protein and also present in our environment. Previous generations amassed vast collections of rocks, papers, photographs, punch cards, microfilm etc. But now it's becoming very difficult for industries to store, retrieve and processes the big data at real time. But now day, the high performance and reliable systems are required to subpart real time work. Map Reduce is a programming model for writing applications that can process Big Data in parallel on multiple nodes. Map Reduce provides analytical capabilities for analyzing huge volumes of complex data.

Keywords: Big Data, Map Reduce, HADOOP, YARN

I. INTRODUCTION

The traditional system of data management is very time consuming and non reliable because of their way of data management. But now day, the high performance and reliable systems are required to subpart real time work. Map Reduce is a programming model for writing applications that can process Big Data in parallel on multiple nodes. Map Reduce provides analytical capabilities for analyzing huge volumes of complex data.

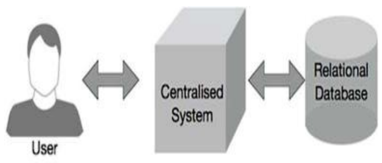

Fig.1 Steps of Data Processing

Now have many effective and efficient applications such as HADOOP, which can run Map reduce program in various languages like JAVA, RUBY, PYTHON etc. Map reduce is a parallel of programming model for processing large data sets or cross the cluster of computers. Maps reduce divides a task into small parts and then assign them to many computers. As it's clear that Mapreduce is a programming model which is used for processing large volume of data.
Mapreduce‘s process uses key and value pairs. The main goal of Mapreduce is automatic parallelization and distribution of tasks. Mapreduce programs are generally written in JAVA but not always other languages are also support. The primary purpose of YARN is to allow for large cluster to scale beyond the four thousand node range. Mapreduce created by Google and it is a fount tolerance which means it can handles failure by re-implementing the failed task on other node. The first international workshop on mapreduce and it’s Application was held in June 2010 in Chicago. The primary differences between big data projects of the present and past are in the form of data takes and how it is put together , presented and analysed, previous generation amassed vast collection of rocks, paper, photo graph , microfilms , punch card , dried plants ,stuffed birds ,But now big data is digital, it’s measured in Exabyte, zetta bytes and yotta bytes or 10^{18} , 10^{21} and 10^{24} bytes respectively.

Manage and process data within a tolerable clasped time. Big data has certain characteristics and hence is defined using 4 V’s namely.

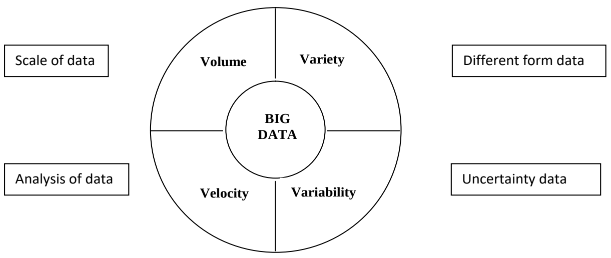

Fig.2 Big Data Characteristics

1) VOLUME: It is the quantity of generated and stored data determines the value and potential insight and whether it can actually be considered big data or not.
2) VARIETY: It is the type and nature of data. This helps to analyse it to effectively use the resulting insight.
3) VELOCITY: In this (velocity) the speed at which the data is generated and processed to meet the demands and challenges that lie in the path of growth and development.
4) VARABILITY: It means inconsistency of the data set can hamper process to handle and manage it or knowing whether the data is available is coming from a credible source is of utmost importance before deciphering and implementing Big data for business needs.

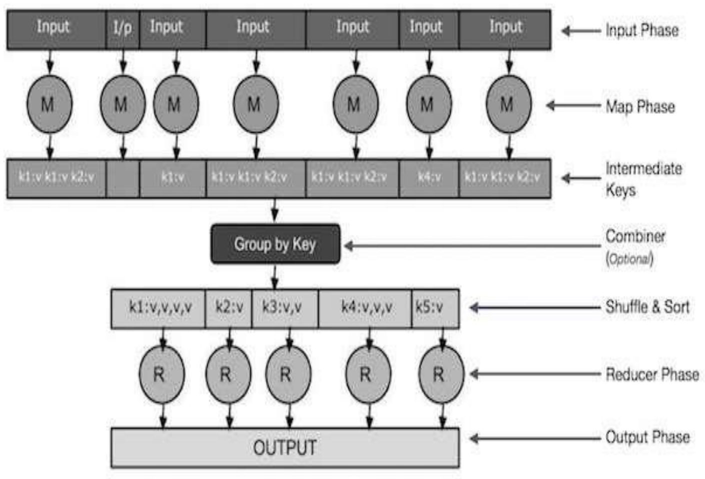

Fig.3. Big Data Phase

Input Phase − Here we have a Record Reader that translates each record in an input file and sends the parsed data to the mapper in the form of key value pairs.

Map − Map is a user-defined function, which takes a series of key-value pairs and processes each one of them to generate zero or more key-value pairs.

Intermediate Keys − The key-value pairs generated by the mapper are known as intermediate keys.

Combiner − A combiner is a type of local Reducer that groups similar data from the map phase into identifiable sets. It takes the intermediate keys from the mapper as input and applies a user-defined code to aggregate the values in a small scope of one mapper. It is not a part of the main Map Reduce algorithm; it is optional.

Shuffle and Sort − The Reducer task starts with the Shuffle and Sort step. It downloads the grouped key-value pairs onto the local machine, where the Reducer is running. The individual key-value pairs are sorted by key into a larger data list. The data list groups the equivalent keys together so that their values can be iterated easily in the Reducer task.

Reducer − The Reducer takes the grouped key-value paired data as input and runs a Reducer function on each one of them. Here, the data can be aggregated, filtered, and combined in a number of ways, and it requires a wide range of processing. Once the execution is over, it gives zero or more key-value pairs to the final step.

Output Phase − In the output phase, we have an output formatter that translates the final key-value pairs from the Reducer function and writes them onto a file using a record writer.

Let us try to understand the two tasks Map & Reduce with the help of a small diagram:

Input Phase − Here we have a Record Reader that translates each record in an input file and sends the parsed data to the mapper in the form of key value pairs.

Map − Map is a user-defined function, which takes a series of key-value pairs and processes each one of them to generate zero or more key-value pairs.

Intermediate Keys − The key-value pairs generated by the mapper are known as intermediate keys.

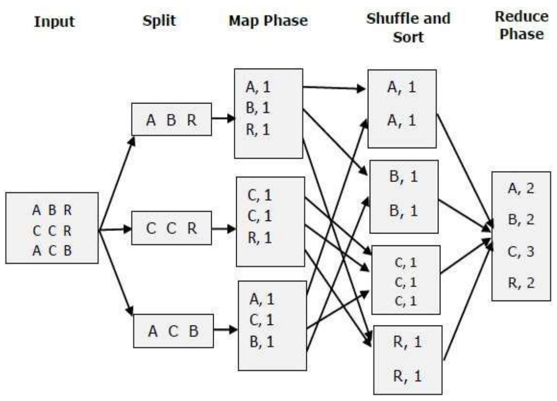

Fig. 4. Map & Reduce Process Diagram

Combiner − A combiner is a type of local Reducer that groups similar data from the map phase into identifiable sets. It takes the intermediate keys from the mapper as input and applies a user-defined code to aggregate the values in a small scope of one mapper. It is not a part of the main Map Reduce algorithm; it is optional.

Shuffle and Sort − The Reducer task starts with the Shuffle and Sort step. It downloads the grouped key-value pairs onto the local machine, where the Reducer is running. The individual key-value pairs are sorted by key into a larger data list. The data list groups the equivalent keys together so that their values can be iterated easily in the Reducer task.

Reducer − The Reducer takes the grouped key-value paired data as input and runs a Reducer function on each one of them. Here, the data can be aggregated, filtered, and combined in a number of ways, and it requires a wide range of processing. Once the execution is over, it gives zero or more key-value pairs to the final step.

Output Phase − In the output phase, we have an output formatter that translates the final key-value pairs from the Reducer function and writes them onto a file using a record writer.
5. The challenges include capture, duration, storage, search, sharing, transfer, analysis and visualization.
6. Big Data requires massively parallel software which could run on tens, hundreds and even thousand of servers.
Map Reduce very useful application and it has been adapted several computing environments such as multicore systems, multi-cluster systems, dynamic cloud environments and high-performance computing environments, but lack of novelty and restricted programming framework are minus points of this application. Some implementations of Map Reduce are Apache Hadoop, Couch db, infinite span, Riak Disco (by Phython).

II. LITERATURE REVIEW

A considerable research work has been done in the field of big data. In previous studies the authors have suggested various methods for catering to the problems in hand through Map Reduce framework over HDFS (Hadoop Distributed File System). Map Reduce is a programming paradigm that runs in the background of Hadoop to provide scalability and easy data-processing solutions.
In the following the researcher has summarized various studies which have been made so far in the literature. Rezgui et.al (2017)[1] have defined Cloud Finder, a system that supports the efficient execution of big data workloads on volunteered federated clouds (VFCs). They have shown two results. First, execution times of the same big data workload at different aggregate locations of a given federation may vary substantially.
Second, Cloud Finder is able to find efficient placements of big data workloads in large cloud federations.
Lim N., Majumdar S. and Ashwood-Smith P. (2017)[2,] studied on improved the robustness of HCP-RM by introducing a mechanism to handle errors/inaccuracies in user estimates of job execution times that are submitted as part of the job’s SLA. Results of experiments conducted on a Hadoop cluster deployed on Amazon EC2 demonstrate the effectiveness of the PSEH technique in improving system performance. The researchers have also been investigates the effect of error in algorithms that depend on user-estimated job runtimes, such as backfilling algorithms and shortest job first.
Susmitha V et al. (2017)[3], studied to improve the availability of HDFS by enhancing the data locality, and contribution focused on many points such as design the replication management system which is truly adaptive to the characteristic of the data access pattern. The approach not only pro-actively performs the replication in the predictive manner, but also maintains the reliability by applying the erasure coding approach. Proposed a complexity reduction method to solve the performance issue of the prediction technique. They also studied on complexity reduction method

significantly accelerates the prediction process of the access potential estimation. They approached to improve the availability while keeping the reliability in connection to the default scheme. Lei Chen1 et.al (2017)[4], studied to design and implementation of size-based scheduling policies in Map Reduce-based systems. Map Reduce workloads provide an interesting job model with intra-job data precedence constraints between the map and reduce phases which can run on any number of resources as long as the input data is accessible through a distributed file system.

Ruiu P., Scionti A., Nider J. and Rapoport M.,(2016)[5], have developed an effective use of heterogeneous architectures in cloud infrastructures. And presented a way of intelligently partitioning applications in such a way that different components can take advantage of the heterogeneous hardware, providing energy efficiency. Finally, the integration of micro services and heterogeneous architectures, as well as the challenge of managing legacy applications, is presented in the context of the OPERA project.

Ghit B. and Epema D. (2016)[6], have developed a statistical model for dynamically setting the runtime limits that achieves near optimal job slowdown performance, and we empirically evaluate TYREX on a cluster system with workloads consisting of both synthetic and real-world benchmarks. Researchers found that TYREX cuts in half the job slowdown variability while preserving the median job slowdown when compared to state-of-the-art Map Reduce schedulers such as FIFO and FAIR. Furthermore, TYREX reduces the job slowdown at the 95th percentile by more than 50% when compared to FIFO and by 20-40% when compared to FAIR.

Zeng X et.al (2016)7, have proposed a novel greedy scheduling algorithm (MASA) that considers the users constraints in order to minimize cost of renting Cloud resources while considering the user's budget and deadline constraints. The simulation results show 25-60% reduction cost in comparison to current methods by using our proposed algorithm.

Nguyen DT-T et al. (2016)[8], have considered the problem of similarity search over the large datasets in the distributed environment. The proposed frame-work employs the Vp-Tree algorithm that integrated on top of the Map Reduce framework to achieve good performance as well as meet the scalability and fault tolerance requirements for the system while data scale up. Researchers proposed a new approach to using it in the parallel environment. The key point of the Vp-Tree algorithm is that it distributed the similar data points into groups, thereby reducing number of data need to scan during the searching stage.

Alam M., Shakil K A. and Sethi S. (2016)[9], have included statistical profile of jobs based on resource usage, clustering of workload patterns and classification of jobs into different types based on k-means clustering. Though there have been earlier works for analysis of this trace, but analysis provides several new findings such as jobs in a production trace are tri modal and there occurs symmetry in the tasks within a long job type.

Gregory A and Majumdar S. (2016)[10], have focused on the matchmaking and scheduling of multi-stage workflows such as Map Reduce which are widely adopted among researchers and industry practitioners for big data analytics. A representative sample of the considerable body of work regarding resource management of Map Reduce workloads both with and without SLAs on cloud environments is presented.

Sangroya A., Bouchenak S. and Serrano D. (2016)[11], presented MRBS, a comprehensive benchmark suite for evaluating the dependability and performance of MapReduce systems. MRBS includes five benchmarks covering several application domains and a wide range of execution scenarios such as data-intensive vs. compute-intensive applications, or batch applications vs. online interactive applications. MRBS allows to inject various workloads, dataloads and faultloads, and produces extensive reliability, availability and performance statistics. They implemented the MRBS benchmark suite for Hadoop MapReduce, and they illustrate its use with various case studies running on Amazon EC2 and on a private cloud.Yang L., Dai Y. and Zhang B. (2016)[12], have presented a Map Reduce scheduling frame work

mitigate performance degradation caused by the performance interference. The framework includes a performance interference prediction module and an interference aware scheduling algorithm. To verify its effectiveness, we have done a set of experiments on a 24-node virtual Map Reduce cluster. The experiments illustrate that the proposed framework can achieve a performance improvement in the virtualized environment compared with other Map Reduce schedulers.

Singhal R. and Verma A. (2016)[13], studied on a simulator based what-if engine to predict job execution time of a MapReduce based application for varying size of cluster with varying types of heterogeneity in the cluster and growing data sizes. The simulator has been validated for three open source MapReduce benchmarks and two industrial Hive based analytic workloads on three different heterogeneous clusters for data sizes up to 100 GB. The largest cluster size considered is of 484 cores, with three types of hardware nodes. They have observed the average prediction error to be within 10% of the actual job execution time.

Zhihong Liu Z et al. (2016)[14], d eveloped a ROUTE, a run-time robust reducer workload estimation technique for MapReduce. ROUTE progressively samples the partition size of the early completed mappers, allowing ROUTE to perform estimation at run time yet fulfilling the accuracy requirement specified by users. ROUTE can achieve high applicability to a wide variety of applications. Through experiments using both real and synthetic data on an 11-node Hadoop cluster, ROUTE can achieve high accuracy with error rate no more than 10.92% and an improvement of 40.6% in terms of error rate while compared with the state-of-the-art solution. Besides, through simulations using synthetic

data. ROUTE is robust to a variety of skewed distributions. ROUTE to existing load balancing and deadline-aware scheduling frameworks and show ROUTE significantly improves the performance of the frameworks.
Shi Y et al. (2016)[15], have analysed the process of job execution and depicts the existing issues why short jobs run inefficiently in Hadoop. According to the characteristic of task execution in multi-wave under cluster overload, they developed a mechanism in light of resource reuse to optimize short jobs execution. This mechanism can reduce the frequency of resource allocation and recovery. Experimental results suggest that the developed mechanism based on resource reuse is able to improve effectiveness of the resource utilization. In addition, the runtime of short jobs can be significantly reduced.
Tang S., Lee B-S. and He B. (2016)[16], proposed two classes of algorithms to minimize the makespan and the total completion time for an offline MapReduce workload. In first class algorithms focuses on the job ordering optimization for a MapReduce workload under a given map/reduce slot configuration. In second class the algorithms considers the scenario that can perform optimization for map/reduce slot configuration for a MapReduce workload. they perform simulations as well as experiments on Amazon EC2 and show that proposed algorithms produce results that are up to 15% „ 80% better than currently unoptimized Hadoop, leading to significant reductions in running time in practice.
Sun M. et al. (2016)[17], have built HPSO (High Performance Scheduling Optimizer), a prefetching service based task scheduler to improve data locality for MapReduce jobs. The basic idea is to predict the most appropriate nodes for future map tasks based on current pending tasks and then preload the needed data to memory without any delaying on launching new tasks. Also implemented HPSO in Hadoop-1.1.2. The experiment results have shown that the method can reduce the map tasks causing remote data delay, and improves the performance of Hadoop clusters.
Sathya V. and Chandramohan K. (2014)[18], focused on scheduling algorithm and technique for managing multi-job Map Reduce workloads that relies on the ability to dynamically build performance models of the executing workloads, and uses the models to provide dynamic performance management using Adaptive scheduler. Map-Reduce framework is mainly based Adaptive scheduler to maximize data locality across working sets, in an attempt to reduce network bottlenecks and increase overall system throughput. Data locality is achieved when data is stored and processed on the same physical nodes. Sometime the server based executing workloads are not delivered to the particulars. Because, the multi-job network areas occurred some problem. So, the server storage is too high. They also overcome the problem by the use of another server that is related to the main server. The problem of main server workload data executing to related server.
Dittrich J. and Quian´eRuiz J-A. (2014)[19], focused on different data management techniques, going from job optimization to physical data organization like data layouts and indexes. They also highlighted the similarities and differences between Hadoop MapReduce and Parallel DBMS.
Rezgui A. and Rezgui S. (2014)[20], have focused on the problem of efficiently allocating this dynamic, heterogeneous capacity to a flow of incoming Virtual Machine (VM) instantiation requests. Researchers have implemented VCFSim, a VCF simulator that uses the proposed resource allocation solution. The results of the experimental evaluation indicated that the proposed approach is able to improve the success rate of VM instantiation requests by up to 38% compared to an approach that uses exact matching with no demand forecasting.
Rezgui A. et al.(2014)[21], presented a stochastic technique that forecasts future demand to efficiently allocate VMs to VM instantiation requests. Also approached on the uses of Markov Chain Monte Carlo (MCMC) simulation known as the Poisson Gamma Gibbs (PGG) sampler. The PGG sampler is used to determine the arrival rate of each type of VM instantiation requests. The arrival rate is then used to determine an optimal VM placement for the incoming VM instantiation requests. They compared approach to a solution that adopts a static smallest-fit approach. The experimental results showed that the solution reacts quickly to abrupt changes in the frequency of VM instantiation requests and performs 10% better than the static smallest-fit approach in terms of the total number of satisfied requests.
Ahmed A E S., Alsammak A K. and Algizawy E. (2013)[22], focused to improve the computing power for the cloud computing platform without charging any extra cost since generic machines are owned by the enterprise. A new Resources management mechanism is introduced to manage the combination of the dedicated and generic machines. In order to implement and achieve the goals of the proposed approach several challenges should be conquered. Open stack cloud computing platform is used and extended in such a way that guarantees QoS and an opportunistic use of the idle or underused generic or public computing resources.
Polo J. et al. (2012)[23], introduced a new task scheduler for a MapReduce framework that allows performance-driven management of MapReduce tasks. The proposed task scheduler dynamically predicts the performance of concurrent MapReduce jobs and adjusts the resource allocation for the jobs. It allows applications to meet the performance objectives without over provisioning of physical resources.
Ardagna D et al. (2012)[24], proposed a supporting system developers and operators in exploiting multiple Clouds for the same system and in design, development and operation method and features a Decision Support System to enable risk analysis migrating the systems from Cloud to Cloud as needed. MODACLOUDS offers a quality driven for the selection of Cloud providers and for the evaluation of the Cloud adoption impact on internal business processes., MODACLOUDS offers a run-time environment for observing the system under execution and for enabling a feedback

loop with the design environment. This allows system developers to react to performance fluctuations and to re-deploy applications on different Clouds on the long term.
Verma A., Cherkasova L. and Kumar Roy H. K V. (2012)[25], introduced and analyze a set of complementary mechanisms that enhance workload management decisions for processing MapReduce jobs with deadlines. The three mechanisms considered as following: 1) a policy for job ordering in the processing queue; 2) a mechanism for allocating a tailored number of map and reduce slots to each job with a completion time requirement; 3) a mechanism for allocating and deallocating (if necessary) spare resources in the system among the active jobs. They analyze the functionality and performance benefits of each mechanism via an extensive set of simulations over diverse workload sets. The proposed mechanisms form the integral pieces in the performance puzzle of automated workload management in MapReduce environments.
Tobias Kurze T et al. (2011)[26], focused on a concept of service aggregation characterized by interoperability features, which addresses the economic problems of vendor lock-in and provider integration approaches challenges like performance and disaster-recovery through methods such as co-location and geographic distribution. The concept of Cloud Federation enables further reduction of costs due to partial outsourcing to more cost-efficient regions, may satisfy security requirements through techniques like fragmentation and provides new prospects in terms of legal aspects. They also discussed architecture that enables new service models by horizontal and vertical integration. The definition along with the reference architecture serves as a common vocabulary for discussions and suggests a template for creating value-added software solutions.
Shams K S et al. (2010)[27], described Polyphony, a resilient, scalable, and modular framework that efficiently leverages a large set of computing resources to perform parallel computations. Polyphony can employ resources on the cloud, excess capacity on local machines, as well as spare resources on the supercomputing center, and it enables these resources to work in concert to accomplish a common goal. Polyphony is resilient to node failures, even if they occur in the middle of a transaction. They concluded with an evaluation of a production-ready application built on top of Polyphony to perform image-processing operations of images from around the solar system, including Mars, Saturn, and Titan.
Celesti A et al. (2010)[28], focused on the authentication agent, which is responsible for a secure federation. They proposed a technical solution based on the IdP/SP model along with the SAML technology. Described an architecture for the federation establishment, where clouds that need external resources ask to federated clouds the renting of extra physical resources. The architecture introduces a new module named Cross-Cloud Federation Manager including three agents (Discovery, Matchmaking and Authentication).
Ananthanarayanan G rt al. (2010)[29], present GRASS, which based on first principles analysis of the impact of speculation. GRASS delicately balances immediacy of improving the approximation goal with the long term implications of using extra resources for speculation. Evaluations with production workloads from Facebook and Microsoft Bing in an EC2 cluster of 200 nodes shows that GRASS increases accuracy of deadline-bound jobs by 47% and speeds up error-bound jobs by 38%. GRASS's design also speeds up exact computations (zero error-bound), making it a unified solution for straggler mitigation.
Polo J et al. (2010)[30], presented a scheduling technique for multi-job MapReduce workloads that is able to dynamically build performance models of the executing workloads, and then use these models for scheduling purposes. This ability is leveraged to adaptively manage workload performance while observing and taking advantage of the particulars of the execution environment of modern data analytics applications.
Condie T et al. (2010)[31], modified version of the Hadoop Map Reduce framework that supports online aggregation ,which allows users to see"early returns" from a job as it is being computed. Hadoop Online Prototype(HOP)also support continuous queries,which enable MapReduce programs to be written for applications such as event monitoring and stream processing. HOP retains the fault tolerance properties of Hadoop and can run unmodified user-defined MapReduce programs.
Zaharia M et al. (2009)32, developed two simple techniques, delay scheduling and copy-compute splitting, which improve throughput and response times by factors of 2 to 10. Although focus on multi-user workloads, techniques can also raise throughput in a single-user, FIFO workload by a factor of 2.
Seo S et al. (2009)33, proposed two optimization schemes, pre fetching and pre-shuffling, which improve the overall performance under the shared environment while retaining compatibility with the native Hadoop. The proposed schemes are implemented in the native Hadoop-0.18.3 as a plug-in component called HPMR (High Performance Map Reduce Engine), evaluation on the Yahoo! Grid platform with three different workloads and seven types of test sets from Yahoo! shows that HPMR reduces the execution time by up to 73%.

III. RESEARCH GAPS

After review of literature, the researcher has figure out the following gaps:
Rezgui et.al (2017), have stated that the cloud federation must scale up when its capacity grows and gracefully scale down when its capacity shrinks. In addition to the aspects of security and scalability, Researchers will focus in future

work on improving the current optimization approach used in Cloud Finder on considering the factors. If the end goal is solely to minimize execution time, then possible factors could be fasted processors, most memory, least co-located VM interference, etc. If the goal is to minimize energy usage, then possible factors could be processor utilization, disk I/O requirements, memory swaps, co-located VM interference, in the total execution time of a big data workload.
Lim N., Majumdar S. and Ashwood-Smith P. (2017), have studied that the superior performance of HCP-RM-EH can be attributed to the PSEH technique being able to make the user estimated task execution times more accurate, which enables HCP-RM-EH to make intelligent matchmaking and scheduling decisions that lead to a high system performance. Even though, the PSEH technique was effective, the resulting adjusted execution times may still not be much accurate.
Thus, for future research, work on devising techniques that perform further error handling at runtime after the jobs start running on the system.
Zeng X et.al (2016), have study that the Big Data is gaining importance, more and more applications have been redesigned to use Big Data frameworks such as Apache Hadoop that supports Map-Reduce programming model. These applications are generally hosted on Public Clouds which provide virtually infinite on-demand storage and computing resources. Researchers have also proposed an important gap concerning scheduling of Map-Reduce jobs on Public Clouds considering while SLA requirements of a user in terms of budget and deadline.
Nguyen DT-T et al. (2016), have presented a searching problem on large data set in the distributed environment. Researchers have been suggested that reducing the response time for each query to make the system can adapt in the near real time environment as well as optimize Hadoop configuration also one of our future works.
Alam M., Shakil K A. and Sethi S. (2016), have conducted analysis of Google cluster trace by providing a statistical profile of workload behavior and clustering of jobs based on resource requirements. Though there have been earlier works for analysis of this trace, but the analysis done in this work provides several new findings such as tri modal behavior of jobs in a production trace and occurrence of symmetry in the tasks within a long job type. Researchers have suggested that this work can be extended in future to create simulators for Google and other similar production traces. The findings can also be validated further by using Hadoop Map Reduce framework on a live production server.
Tang S., Lee B-S. and He B. (2016), have focused on the job ordering and map/reduce slot configuration issues for Map Reduce production workloads that run periodically in a data warehouse, where the average execution time of map/reduce tasks for a Map Reduce job can be profiled from the history run, under the FIFO scheduling in a Hadoop cluster. Two performance metrics are considered, i.e., makespan and total completion time.

They are also proposed several future works such as:
(1) Further consider map/reduce slot configuration issues for online jobs under FIFO scheduling.
(2) To consider other existing Hadoop schedulers and Capacity scheduler.
(3) Focused on more complex workloads of jobs with precedence dependency, where the constraint job ordering approach will be considered.

Xiaoyi Lu et al. (2016), have present a detailed design for high performance RDMA-based Apache Spark on bare-metal and SR-IOV enabled Infinite Band clusters. In the future, to investigate further on the architectural bottlenecks of Spark and propose more designs to accelerate Spark workloads on modern HPC clusters.
Singhal R. and Verma A. (2016), The proposed MR simulator collects map and reduce tasks measurements on a small cluster with low data volume. Based on this and linear regression, researchers can predict the job performance for any large cluster with mix of heterogeneous set of machines and any large data size. Researchers have validated the what-if engine for open source benchmarks and industrial applications in financial and Telecom domain. These applications' performance have been predicted in mix of large, medium and small configuration machines with up to 484 cores in the cluster and 100 GB as largest data size. They observed the model to be accurate with an average prediction error less than 10% in all cases.
In future, to validate the model for different data distributions and different settings of MR framework for much larger larger data and cluster sizes. They plan to extend the what-if engine for estimating job execution time with different MR framework configurations as input. This will help in predicting the best performance of an application for large data size in larger heterogeneous cluster – we will do so by including optimal tuning of the application on the projected cluster.

IV. CONCLUSION

The review of literature encompassing research colloquium, conference proceedings and research papers have been carried out by the scientists to have a deeper insight into management of bigdata and parrellal applications like Mapreduce. If any company properly plan for success get a head of data quality and ownership challenges , proactively address, company's talent sources and management strategies then the company could manage its big data properly. Company or organization need mix of well trained staffs, and contingent Labour to address various elements

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

Vol. 7, Issue 12, December 2018

as the company's plans, build and runs it's Big Data projects. Many applications of mapreduce such as Hadoop DB, SQL, Server 2005,and etc. Some applications exhibited significant efficiency advantage than Hadoop mapreduce when processing various amount of data sets.
Map reduce the one of the most important framework in distributed cloud computing. The drawback of mapreduce is workload scheduling job label or task level are the traditional methods of schedule workload at coarse-grained levels. Hadoop and the associated Mapreduce have become the important platform for the cost effective analytics over big data.

REFERENCES

[1]. Rezgui A., Davis N., Malik Z., Medjahed B. and Soliman H., “CloudFinder: A System for Processing Big Data Workloads on Volunteered Federated Clouds”, IEEE 2017, pp. 1-12, DOI 10.1109/TBDATA.2017.2703830.
[2]. Lim N., Majumdar S. and Ashwood-Smith P. "Techniques for Handling Error in User-estimated Execution Times During Resource Management on Systems Processing MapReduce Jobs", IEEE, 2017, pp.788-793, DOI 10.1109/CCGRID.2017.70.
[3]. Susmitha V., Poojitha P., Aasritha Ch., Lavanya V. and Sagar V. V, "Flexible Replication Management for Frequently Accessed Data Files in HDFS Using Hadoop", IJMTST, Vol. 3, 2017, pp. 24-31.
[4]. Chen L., Zhang J., Li-Jun C., Zi-Yun D and Meng T. "A Cross-Jobs-Cross-Phases Map-Reduce Scheduling Algorithm in Heterogeneous Cloud" Journal of Computers Vol. 28, No. 2, 2017, pp. 145-164, http://doi:10.3966/199115592017042804011.
[5]. Ruiu P., Scionti A., Nider J. and Rapoport M., "Workload Management for Power Efficiency in Heterogeneous Data Centers" IEEE, 2016, pp.23-30, DOI 10.1109/CISIS.2016.107.
[6]. Ghit B. and Epema D., "Tyrex: Size-based Resource Allocation in MapReduce Frameworks", IEEE, 2016, pp. 11-20, DOI 10.1109/CCGrid.2016.82.
[7]. Zeng X, Garg S. , Wen Z., Strazdins P., Wang L. and Ranjan R., "SLA-aware Scheduling of Map-Reduce Applications on Public Clouds", IEEE, 2016, pp/ 655-662, DOI 10.1109/HPCC-SmartCity-DSS.2016.183.
[8]. Nguyen DT-T., Yong C. H., Pham X-Q., Nguyen H-Q., Loan TTK. and Huh E-N., "An Index Scheme for Similarity Search on Cloud Computing using MapReduce over Docker Container", IEEE, 2016, DOI:http://dx.doi.org/10.1145/2857546.2857607.
[9]. Alam M., Shakil K A. and Sethi S. "Analysis and Clustering of Workload in Google Cluster Trace based on Resource Usage", IEEE, 2016, pp. 740-747, DOI 10.1109/CSE-EUC-DCABES.2016.271.
[10]. Gregory A and Majumdar S., "Energy Aware Resource Management for MapReduce Jobs with Service Level Agreements in Cloud Data Centers", IEEE, 2016, pp. 768-577, DOI 10.1109/CIT.2016.42.
[11]. Sangroya A., Bouchenak S. and Serrano D., "Experience with Benchmarking Dependability and Performance of MapReduce Systems", ELSEVIER, 2016, pp. 1-24.
[12]. Yang L., Dai Y. and Zhang B., "MapReduce Scheduler by Characterizing Performance Interference", China Communications, 2016, pp. 253-262.
[13]. Singhal R. and verma A., "Predicting Job Completion Time In Heterogeneous MapReduce Environments", IEEE, 2016, pp.17-27, DOI 10.1109/IPDPSW.2016.10.
[14]. Zhihong Liu Z., Zhang Q., Boutaba R., Liu Y. and Gong Z., "ROUTE: run-time robust reducer workload estimation for MapReduce", Internation Journal of Network Management, 2016, pp. 224-244, DOI: 10.1002/nem.1928.
[15]. Shi Y., Zhanga K., Cuia L., Liua L., Zhenga Y., Zhanga S. and Yub H., "MapReduce Short Jobs Optimization Based On Resource Reuse", Microprocessors and Microsystems, 2016, doi: 10.1016/j.micpro.2016.05.007.
[16]. Tang S., Lee B-S. and He B., "Dynamic Job Ordering and Slot Configurations for MapReduce Workloads", IEEE, 2016, pp. 1-14, DOI 10.1109/TSC.2015.2426186.
[17]. Sun M., Zhuang H., Zhou X., Lu K. and Li Ch., "Scheduling Algorithm based on Prefetching in MapReduce Clusters", Applied Soft Computing, 2016, pp. 1-28, http://dx.doi.org/doi:10.1016/j.asoc.2015.04.039.
[18]. Sathya V. and Chandramohan K., "Deadline Based Map-reduce Workload Management in Multijob", International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 2, 2014, pp. 5609-5613.
[19]. Dittrich J. and Quian´eRuiz J-A., "Efficient Big Data Processing in Hadoop MapReduce", Information Systems Group 2014.
[20]. Rezgui A. and Rezgui S., "A Stochastic Approach for Virtual Machine Placement in Volunteer Cloud Federations", IEEE, 2014.
[21]. Rezgui A., Quezada G., Rafique M M. and Malik Z., " A Capacity Allocation Approach for Volunteer Cloud Federations Using Poisson-Gamma Gibbs Sampling" IEEE, 2014.
[22]. Ahmed A E S., Alsammak A K. and Algizawy E., "A New Approach to Manage and Utilize Cloud Computing Underused Resources", Volume 76 – No.11, 2013, pp. 29-36.
[23]. Polo J , Carrera D., Becerra Y., Torres J. and Ayguad E., "Deadline-Based MapReduce Workload Management", Vol. 10, No. 2, 2013, pp. 231-244, DOI:10.1109/TNSM.2012.122112.110163.
[24]. Ardagna D., Nitto E D., Casale G., Petcu D., Mohagheghi P., Mosser S., Matthews P., Gericke A., Ballagny C., D'Andria F., Nechifor C S. and Sheridan C., "MODACLOUDS: A Model-Driven Approach for the Design and Execution of Applications on Multiple Clouds", IEEE, 2012.
[25]. Verma A., Cherkasova L. and Kumar Roy H. K V., "Two Sides of a Coin: Optimizing the Schedule of MapReduce Jobs to Minimize Their Makespan and Improve Cluster Performance", IEEE, 2012, pp. 11-18, DOI 10.1109/MASCOTS.2012.12.
[26]. Tobias Kurze T., Klemsy M, Bermbachy D., Lenkz A., Taiy S. and Kunze M., "Cloud Federation", IEEE, 2011.
[27]. Shams K S., Powell. M W., Crockett T M., Norris J S., Rossi R. and Soderstrom T., "Polyphony: A Workflow Orchestration Framework for Cloud Computing", IEEE, 2010.
[28]. Celesti A., Tusa F., Villari M. and Puliafito A., "Three-Phase Cross-Cloud Federation Model: The Cloud SSO Authentication", IEEE, 2010.
[29]. Ananthanarayanan G, Hung M C-C, Ren X., Stoica I., Wierman A. and Yu M., "Reining in the Outliers inMap-Reduce Clusters usingMantri", IEEE, 2010.
[30]. Polo J., Carrera D., Becerra Y., ¸ Torres J., Ayguad E., Steinder M. and Whalley I., "Performance-Driven Task Co-Scheduling for MapReduce Environments", IEEE, 2010, pp. 373-380.
[31]. Condie T, Conway N., Alvaro P. and Hellerstein J M., "MapReduce Online", IEEE, 2010.
[32]. Zaharia M., Borthakur D., Sarma J S., Elmeleegy K, Shenker S. and Stoica I., "Job Scheduling for Multi-User MapReduce Clusters", IEEE, pp. 1-16.
[33]. Seo S., Jang I., Woo K., Kim I., Kim J-S. and Maeng S., "HPMR: Prefetching and Pre-shuffling in Shared MapReduce Computation Environment", IEEE, 2009.