International Journal of System Dynamics Applications
Volume 10 • Issue 4 • October-December 2021

An Approach in Big Data Analytics to Improve the Velocity of Unstructured Data Using MapReduce

Sundarakumar M. R., AMC Engineering College, Bengaluru, India
Mahadevan G., AMC Engineering College, Bengaluru, India
Ramasubbareddy Somula, VNRVJIET, Secunderabad, India
Sankar Sennan, Sona College of Technology, Salem, India
Bharat S. Rawal, Gannon University, USA

ABSTRACT

Big data analytics is an innovative approach to extract the data from a huge volume of data warehouse systems. Hadoop is a framework, which is used to perform high speed data retrieval from various clusters by MapReduce and HDFS methods. The huge volumes of files are accessed using data mining, machine learning, and deep learning algorithms. However, these techniques take more time to retrieve the data among the clusters. To overcome the latency issue, the proposed work applies the hybrid algorithm, namely compressed elastic search index (CESI) and MapReduce-based next generation sequencing approach (MRBNGSA), in scheduling and shuffling phase. This proposed approach provides the tangible changes over the MapReduce phases. The performance of the proposed CESI-MRBNGSA algorithm provides significant performance than Hadoop BAM and GATK.

KEYWORDS

Big Data, CESI, MapReduce, MRBNGS

1. INTRODUCTION

In this era big data makes a new revolution in day to day activities of social media, health care, banking sector, Military division, and industries. But the problem of big data reveals in the form of accessing their volume, variety, and velocity. Because now a day's data generated by humans as well as machines controlling is not an easy job with old techniques. The place where it has a lot of formats deals with a major problem. Moreover, the speed of the data retrieval and accessing from the data warehouse seems tremendous challenges and issues for stream processing. Big Data can be processed like batch, periodic, Near to real-time and real-time makes conflict on cluster configuration. Batch processing doesn't support iterative and multi pass operations. Digital data like structured, semi-structured and unstructured formats storage is typical challenge environment. While extracting the data from those clusters the time of retrieval is high using a map-reduce method. All the input sends as Single Pass which means a set of smaller files group. Here the issues are multiple passes and real-time data integration is not possible in map-reduce for data processing using old methods. Hadoop Distributed File System storage allows a huge volume of data storage in the form of scale-

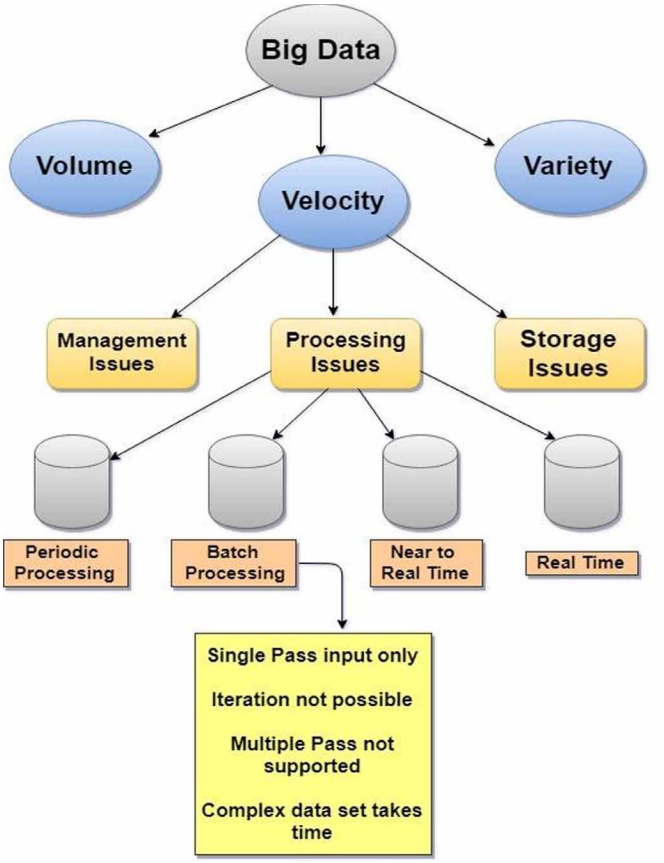
DOI: 10.4018/IJSDA.20211001.oa6

out architecture. The following Figure 1 shows the challenges and issues in big data while doing the data processing.

To solve the problem of latency delay in big data many tools and frameworks have been used like Apache Hadoop, Apache Spark. However, Map Reduce and HDFS methods are used to provide solutions for the data retrieval problem in big data sets (McCreadie et al., 2012). But the Map-Reduce phase has taken more time to complete multiple jobs. Because of that, store and retrieve data from the HDFS is also taking time. Most of the data mining concepts and algorithms are provided different solutions to cover this problem. Perhaps, it provides near to real-time processing of data. When real-time data retrieval scenario comes, none of the techniques give solutions for time consumption in data retrieval. Eventually, big data analytics can be done by CAP (Consistency, Availability, and Partition) theorem and Shared Nothing Architecture (SNA) (Duggal & Paul, 2013). When big data can be processed by Map Reduce concept Machine Learning Algorithms are used to segregate the tasks based on its Metadata. The complete tasks in the Map phase is divided into smaller tasks and it will be processed by mapper () function (Alfonseca et al., 2013). Separate keys are assigned to the tasks given by the clients and that has to be shared among the mapped function elements. Different algorithms (Velusamy et al., 2013), (R. Somula & Sasikala, 2019) were used to share those keys with computational logic and other concepts for security purposes. Using these scenarios data processing in HDFS with Map Reduce concepts is very clumsy. Data mining techniques association rules, K means, and Nearest Neighbor cluster algorithms and other concepts are not provide real-time data retrieval processing. Apache Spark will provide a solution for real-time data processing with in-memory analytics techniques. In Apache Spark, both databases and data warehouse engines are located on the same block so that it can perform very fast when compared with ancient techniques (R. Somula et al., 2019).

In big data, security will be the major problem for keeping all the data in cluster(Weets et al., 2015), (Remya et al., 2018). Generally, big data keep track of all data inside the cluster based on the dynamic resource demand techniques. Suppose nodes are removed from the cluster due to node failure, it has to reallocate resources virtually by Net Merger protocols (Yu et al., 2013), (Zhang et al., 2015),(Sennan et al., 2019). All these processes are optimizing the time consumption over map-reduce. Map Reduce will give the data periodically, near real-time or real-time based on the purpose of client requests (Luna et al., 2016), (R. Somula & Sasikala, 2018). Machine Learning, Genetic algorithms and decision trees (Xia et al., 2016) are using to solve that issue for data extraction. The time delay to complete the reduce function is less when compared with the map () phase using virtual shuffling and shuffle (Guo et al., 2016), (R. S. Somula & Sasikala, 2018) methods. The shuffling, sorting phases are the main functions of the map-reduce program has a lot of challenges and issues in time delay. During the shuffling phase, CPU utilization and memory can be fully occupied in the resource manager. It is resolved by JVM (Java Virtual Machine) for memory aware techniques. This can control by fine grain and PUMA (Wani & Jabin, 2018) algorithms. Time delay from mapper to reduce phase in map-reduce cannot be solved using memory management and classification techniques (Wani & Jabin, 2018), (Khan et al., 2018),(Li et al., 2018). Next-Generation Sequencing (NGS) (Yang et al., 2018) are the solutions in map-reduce function to solve the issues in alignment of tasks. Perhaps, the sequencing of jobs assigned by job trackers in Hadoop versions can be controlled by Burrows-Wheeler Transform (Samaddar et al., 2018), (Sankar, Srinivasan, Luhach, et al., 2020) and Cloud Burst (Mondal & Khatua, 2019) using array methods for the optimization. Elastic Search is an innovative method for retrieving data quickly from huge data set elements as sub data sets. An inverted index is used to store Metadata of the nodes using tokens and ids assigned for jobs. Indexing based on the position and number of occurrences of each letter in each word and sentence of the entire paragraph. Multidimensional analysis and Feature Nearest Neighbor System (FNNS) (Sundarakumar & Mahadevan, n.d.) is used to finding the distance between different clusters based on its Hamming distance. Simultaneously, duplication and repetitions are removed in the clusters and

Figure 1. Big Data challenges and Solutions



its mapping time of data is very low when compared with previous techniques (Sankar, Srinivasan, Ramasubbareddy, et al., 2020).

Map Reduce is the programming model that contains several phases like splitting, mapping, shuffling, sorting, partitioning, combining and reducing. When the file is getting as an input from different client's keys are generated based on the job size of the input file. Mapper () and Reduce () function will give the output file that is given input to the from HDFS data set. There are so many implementations are available in the Hadoop framework for data processing. The following Table 1 and Table 2 denote the techniques provided by other than map-reduce concepts and indicate the applications of map-reduce.

2. LITERATURE SURVEY

(McCreadie et al., 2012) posed methods for handling structured, unstructured and semi-structured data accessing in multiple machines are not effective due to different formats. In that situation, an inverted index was helped to arrange different data but cannot be handled if data is huge. Moreover, different tasks are coming from various machines, indexing scale of each task was not handled in multiple machines during the inverted index are offline. For further processing of information extracting from huge databases, map-reduce program paradigm. The open-source framework called Apache Hadoop is used to perform map-reduce functions with java programming initially. Later all the information

Table 1. Map Reduce Implementation methods

Map Reduce Methods	Pros	Cons
Google Map Reduce	Make duplicate data blocks on multiple nodes for fault tolerance	Batch based architecture not suitable for real time data access
Hadoop	Scalability and Availability	Cluster Management is difficult
Grid Grain	Sub task distribution and load balancing	Does not support non-java applications
Mars	Massive Thread Parallelism in GPU	Not for atomic operations due to expensive
Tiled-Map Reduce	Convergence and Generalization	Cost is high
Phoenix	Multicore CPU	Scalability is less
Twister	Tools are used effectively	Not possible to break huge data set

which was processed by the map-reduce framework will be stored in HDFS (Hadoop Distributed File System)(Sennan, Somula, et al., 2020). Eventually, all functions and operations were performed by the old method called an inverted index. Because of the huge volume of data, it was not in order while sending it as a single pass input file. The inverted index provides tokens and ids for each task by posting a list. This method is used to perform distributed database concepts effectively (Sennan, Ramasubbareddy, et al., 2020).

(Duggal & Paul, 2013), proposed suggestions for accessing all digital data using web indexing and log analytics. Instead of giving a scale-in storage method as a horizontal approach, they suggest a scale-out method for handling a huge volume of data during processing time. In batch processing, byte stream abstraction is not possible in multi node clusters. Moreover, extracting those data from the database engine old techniques OLTP, SQL is taking a lot of time to recover data in a specific format. (Alfonseca et al., 2013) proposed map reduces method for handling data crunching and data backup problems during parallel database distribution. When the cluster has created based on the size of the data and formats there may be a risk in node and task failure. To avoid this during map-reduce

Table 2. Map Reduce Applications

Map Reduce Applications	Pros	Cons
Distributed Grep	Data analysis is generic	Less response time
Word Count	Massive document collection of occurrences	Limited only
Tera Sort	Load balancing in large clusters	Overhead
Inverted Index	Collection of unique posting list	Lots of pairs in shuffling & sorting
Term Vector	Host analysis search	Sequential tasks
Random Forest	High scalability	Low
Extreme Learning Machine	Convergence and Generalization	Uncertainty
Spark	Data fit in memory	Huge memory needed
Algorithms	Data intensive applications	Time consuming
DNA Fragment	Parallel Algorithm	Large memory
Mobile sensor data	Extracting data is easy	Difficult to implement
Social Networks	Quick response	Need more techniques for analysis

programming in Hadoop inverted index is used to find out the tasks given by the users from anywhere in the world. Machine Learning algorithms are used to perform replication among the clusters. To improve the speed of this system setup page rank computation and random walk algorithms are used to perform time minimization by reduced keys in mapper () function.

(Velusamy et al., 2013) proposed scheduling methods for solving virtual machines location identification problem. It has created a complex structure like trees and indexes to find out the location of data. To overcome this issue Hash tables are used in indexing for replication of data between the clusters. All the input data has to store in indexes by FIFO (First in First Out) method. Their implementation will be done by merging those indexes in different places and compressed those in a proper method. Once it is finished it is easy to recover data from anytime, anyplace and anywhere. Moreover, data can be Write Once and Read Many (WORM) times from the cluster in different places.

(Weets et al., 2015) proposed shuffling algorithms for scalability, availability, task scheduling, of map-reduce programming. But the time consuming is not up to the mark because of the volume of data in the cluster. Pipeline concepts are used to get multiple jobs at a time in the map to reduce programming but predictive scheduling is not effectively working because of the keys used in mapper and reducer function. To overcome this situation, Kerberos is a standard that is running in this system for providing security over the keys. Algorithms like LIBRA and PRISM are used in High-Performance Computing used in mapper functions by creating buffers at different places among the clusters.

(Yu et al., 2013), proposed protocol for Map-reduce data movement and reduce I/O device usage in a cluster. To do this Near-demand merging, and dynamic balanced Merging sub trees are used to get required data within a minimum of time. To provide a stream accessing of files between the clusters it uses MOF supplier (map op file) concepts and Net Merger network protocols.(Zhang et al., 2015) proposed approaches for resource utilization in multiple job applications from various clients. But optimizing those jobs can do by giving input to the Hadoop framework through slot-based input system. Map Reduce phase level scheduler concept is used to create awareness about resources that are available inside the cluster. Fine-Grained method is also used to find out the resources that are placed inside the cluster. For implementing this there are two different approaches are used namely RAS-resource aware scheduling for resource utilization and dominant resource fair queuing (DRFQ) to create a queue for a list of resources in the clusters.

(Luna et al., 2016) proposed the mining rules for reorganizing of data sequencing and access with speed in the Hadoop framework. To find the distance between the clusters, it used Hamming Distance and create a combination of data relevancy it used ARM(Association Rule Mining) then to find the mean and standard deviation it used the Apriori algorithm also it used FP growth algorithm for cluster integration. Run-length encoding and inverted index concepts are used to provide the solutions for this above-said problem and it will be used only inside the cluster.(Xia et al., 2016) proposed system and analysis method for heterogeneity nodes and autonomous sources with HACE theorem and real-time prediction system (RPS) techniques. Because huge data can be classified by various data mining algorithms but also it is not possible to store in the appropriate cluster that has many problems. Traffic between the clusters has so many problems by adjusting the features of traffic _flow prediction using correlation analysis (TFPC). It will provide distributed data over the network without any traffic. For optimization nearest neighbor algorithm is used to solve this issue and it will be processed by the map-reduce programming paradigm.

(Nabavinejad et al., 2016) proposed machine for mnemonic architecture processor memory slot problem and finding the number of maps and tasks for allotted task completion. To overcome this situation so many techniques can work under map-reduce concepts like Fine Grain, PUMA, Memory Aware Tuner Classification, Word Count, and Inverted Index, Self-Join. But it will not provide a good solution to solve that issue. So that Shuffle Watcher is used to effectively monitor the JVM and full memory, CPU utilization has done using the bypassing method. (Guo et al., 2016) proposed methods for working with multi-user cluster workload distribution problem because of data skew and scheduling. It can be done by Shuffle manager, Task Scheduling, Longest Approximate Time to

End (LATE), FLEX methods multimode cluster workload balancing. To implement this technique in an effective way it is used shuffle concept by shuffle on write method.

(Wani & Jabin, 2018) proposed techniques for node locality overlapping problem because of imbalanced workload, due to this reduce completion time is more when compared with other techniques. It can be achieved by the techniques Locality-Enhanced Load Balance (LELB) Map, and Local reduce Shuffle and final Reduce (MLSR) phases. It can be further developed by different innovative approaches like Locality-Based Balanced Schedule (LBBS) and Overlapping-Based Resource Utilization (OBRU) for developing an imbalanced workload distribution.

(Khan et al., 2018) proposed new approach for data policy and analysis services for visualization complication because of the scattered distribution of data locality. To solve this problem it used elastic search in multidimensional analysis method over clusters. To implement this Conditional Random Field (CRF) is used to find exact matches from the input given by the user. (Li et al., 2018), Proposed algorithm for the alignment of the output maps from the mapper into the reduce part. To solve this issue, alignment algorithms like Seed and Extend Algorithm, Burrows-Wheeler Transform is used to send the inputs by order. For further implementation, Stream Aligner is used to perform alignment operations using occurrences and matches.

(Yang et al., 2018) proposed scheduling algorithm for Pipeline concepts in the Hadoop environment. For that it uses DAG (Direct Acyclic Graph), Page Rank Algorithm and Naïve Bayes classifications are used to accessing multiple jobs. For future enhancement, it will use Pipeline Improvement Support with Critical chain Estimation Scheduling (PISCES) for accessing multiple jobs in the Hadoop framework.

(Samaddar et al., 2018) proposed new scheduling method for Massive Parallel Sequencing of the input file. To provide the solution for sequencing it is used Next Generation Sequencing (NGS) for the development of the alignment process. It has chosen the same model of genomic pattern from the human DNA structure and the same concept is used to identify the Meta data of the input file. For further development, it is using modern tools named HADOOP-BAM and GATK for parallel sequencing.

(Rexie & Raimond, 2019) proposed algorithms for huge data set retrieval problems and Mapping of two different data sets are the issues in the Hadoop Map Reduce framework. It can be solved by NGS and BWT using alignment methods. Suitable pair got selected from the multiple inputs to do exact pattern selection for the right output with in quick time. For further development of other implementations Stream Aligner, Cloud Burst, Suffix Array (SA) construction methods are used to get data from huge data retrieval. (Mondal & Khatua, 2019) proposed new approach for data alignment based on the incomplete tasks in different clusters. For solving this problem it has distributed pair wise sequence alignment technique called MRaligner is used. For future development, it uses the Smith-Waterman (SW) method to provide proper alignment.

(Sundarakumar & Mahadevan, n.d.) proposed solutions for Cluster classification and duplication of data problem of the distance between the clusters. The mapping time also has the problem. These are all solved by Hamming Distance, Nearest Neighbor, K Means algorithms. But for future enhancement, it will be used Elastic Search, and Feature Nearest Neighbor to predict the cluster correctly and share the data among them.(Wang et al., 2016) proposed to increase the speed of the storage distribution over huge network is based on the sub set creation of data sets during transmission time. It will reduce the time to complete the task based on the size of the task given as input.

(Wang et al., 2016) proposed algorithms to create Cloud framework for business applications while handling big data from large industries. The cluster created in this system deals with HDFS cluster features in order to retrieve huge data set. (Auxilia et al., 2020), proposed model for Sustainable development using data mining algorithms for the extraction of huge data set from different data warehouses on industries., (Zelinka & Amadei, 2019), proposed model for Climate change monitoring using big data analytics by HDFS and Map Reduce concepts. The real time generated climate change data has stored in their developed model for data processing using scheduling algorithms. (Majhi,

2018), (Dutta, 2017) proposed the algorithms to search images from huge data base of breast cancer images and medical diagnosis images in health care sector. The entire data set has accessed using data mining algorithms and associated according to the user inputs. (Elfouly et al., 2017) proposed a algorithm to find Multi object tracking in Transportations of metropolitan cities. The huge crowd of people and vehicles are tracked by multi object system and find the exact matches using pattern recognition and object classification.

3. BACKGROUND AND MOTIVATION

In Hadoop, so many clients are sending their jobs for performing tasks. This can be handled by Job Tracker or resource Manager by Hadoop. There are two different versions are available in Hadoop named as Hadoop 1.X and Hadoop 2.X.Here X denotes the version releases/updates. If Hadoop 1.X used in the cluster, then the tasks can be controlled by the Job Tracker /Resource Manager. If it will be Hadoop 2.X, it may use the secondary name node which is the replica of the Name Node and will be used for copying Metadata from the cluster.

There are three main schedulers are available in Hadoop.

1. FIFO Scheduler
2. Capacity Scheduler
3. FAIR scheduler

3.1 FIFO Scheduler

It is the default scheduler used by Job Tracker and will be effective for the arrangement of tasks in the Map-Reduce paradigm. Jobs are queued in the priority queue and send it to the job tracker. When a job is scheduled, even its priority is lower, no preemption is allowed. So some high priority processes may wait for a long time. The below diagram denotes the functionality of the FIFO scheduler in Hadoop 1.X.

3.2 Capacity Scheduler

It is the default scheduler for Resource Manager in Name Node of Hadoop 1.X.In the Capacity scheduler, there are multiple queues to schedule the tasks. For each queue, there are dedicated slots in the cluster. When no jobs are running, the task of one queue can occupy as many slots as possible. When a new job comes in the next queue, it will replace the slots from those slots which are dedicated to that queue. The below diagram denotes the functions of the capacity scheduler.

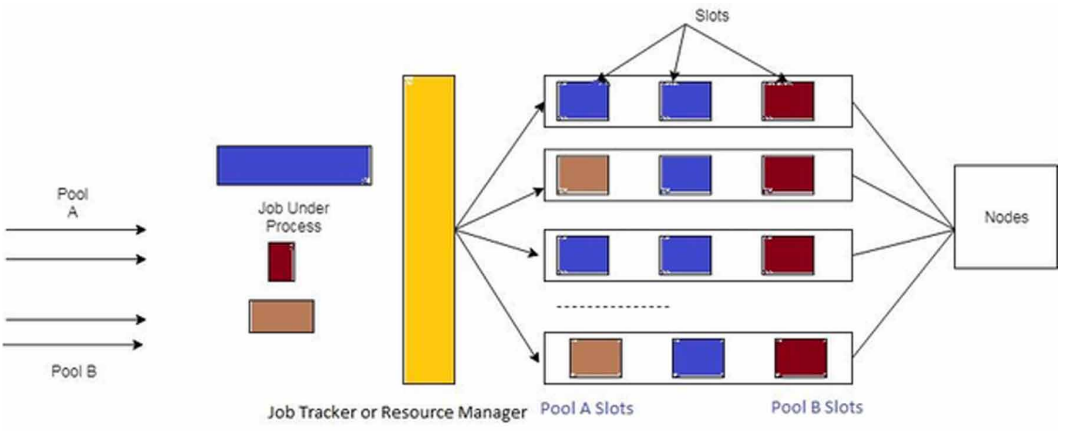
3.3 Fair Scheduler

It is similar to the capacity scheduler. When some high priority jobs are coming in the same queue, it is processed in parallel by replacing some portions of the task from dedicated slots. The below diagram denotes the functions of the Fair scheduler. The following Figure 2 will indicate the working principle of Fair scheduler.

Moreover, a lot of schedulers are used in Map Reduce programming for increasing the speed of job completion time in various aspects. But factors like size of data, length of the job, the number of short reads change the total time to complete the job also changes. The following Table 3 reveals the advantages and disadvantages of various scheduling algorithms work done in different clusters and its remarks.

There are so many schedulers working in map-reduce programming for getting the output within a particular time. But all the systems are getting so much of delay in latency and throughput. Finally, Next Generation Sequencing (Rexie & Raimond, 2019) is a technique to give output data at a certain time while the map reduces the model working in Hadoop multi-node cluster. It is working on Massive

Figure 2. Working of Fair Scheduler



Parallel Sequencing of jobs or tasks given as an input to the map-reduce model and then it will be given output as client requirements.

Before that Elastic Index Search (Sundarakumar & Mahadevan, n.d.) is used to optimize the output of the mapper by means of inverted index techniques. But the values are not exact when it did the experiment. For that, it has to be extended using the Metadata of the given data from the mapper and it will be stored on the elastic index patterns. The method of elastic index search is to avoid the duplication and repetition of the letters from the word or sentence in a paragraph. The cleaned data will be stored in an index that will extend up to the size of the memory occupied by the job.

The above Figure 3 represents the working principle of the elastic search index. Here each and every node wants to extend the usage of Metadata or replica from the master node. In this approach, horizontal partitioning will take care of the storage of databases as a set. Sharding is the concept of creating a subset from the given data set and take a copy of that then stored it in a next node. Respectively the next node sharding and replica will be stored in the previous node. Such a way that single point or multi-node point failure it will be working without any delay. Thus the throughput and latency of the entire Hadoop Ecosystem will be working effectively.

4. THE PROPOSED CESIMRBNGSA -ARCHITECTURE

The proposed system architecture will give a new innovative method or approach to the big data scientist because always data scientists have a problem in extracting data from a huge data set. Because in map-reduce it has a splitter, mapper, shuffle, sort, combiner, reducer parts are there. The time taken to get that data from the data warehouse takes minimal delay with latency due to passing these many steps in a single pass input smaller file. At that time the scientists are concentrated on writing a good algorithm or logic to improve efficiency. This method we will introduce two new concepts called Compressed Elastic Search Index (CESI) and Map Reduce- Based Next Generation Sequencing Approach (MRBNGSA) for the quick retrieval of data from PB servers. In Multi-node clusters, the problem of data extraction is to get the exact data that we have given as a query, but what we got is moderately changed due to their Metadata. Due to this, our new CESI method will give index as a database of Metadata of all original data. It will be insisted on the data cleaning process which will be done by Elastic Search Index then it will remove the duplication from the input. Moreover, all input files may not have Metadata, such cases repetition of the letter can be considered for the cleaning process and it would be cleared by removing it in the entire paragraph. Likewise, all the duplicated data has removed and given it to the next phase of map-reduce named shuffle and sort. This part

Table 3. Various Scheduling Algorithms and remarks

Scheduler	Advantages	Disadvantages	Remarks
FIFO Scheduler	Easy Implementation	Bad data locality	Static Allocation
FAIR Scheduler	Short response time	Unbalanced workload	Homogeneous System
CAPACITY Scheduler	Unused Capacity jobs	Complex implementation	Homogeneous System, Non primitive
Delay Scheduler	Simplified Scheduling	Not work in all situation	Homogeneous System, Static
Matchmaking Scheduler	Good Data locality	More response time	Homogeneous System, Static
LATE Scheduler	Heterogeneity	Lack of reliability	Homogeneous System & Heterogeneity
Deadline Constraint Scheduler	Timing and optimization	Cost is high	Homogeneous System, Heterogeneity, Dynamic
Resource Aware Scheduler	Cluster Resource Monitoring	Extra time for monitoring	Homogeneous System, Heterogeneity, Dynamic
HPCA Scheduler	High hit rate and redundancy	Cluster change state	Homogeneous System, Heterogeneity, Dynamic
Round Robin Scheduler	Proper work completion	No priority is given	Homogeneous System, Heterogeneity, Dynamic

Figure 3. Elastic Search Index

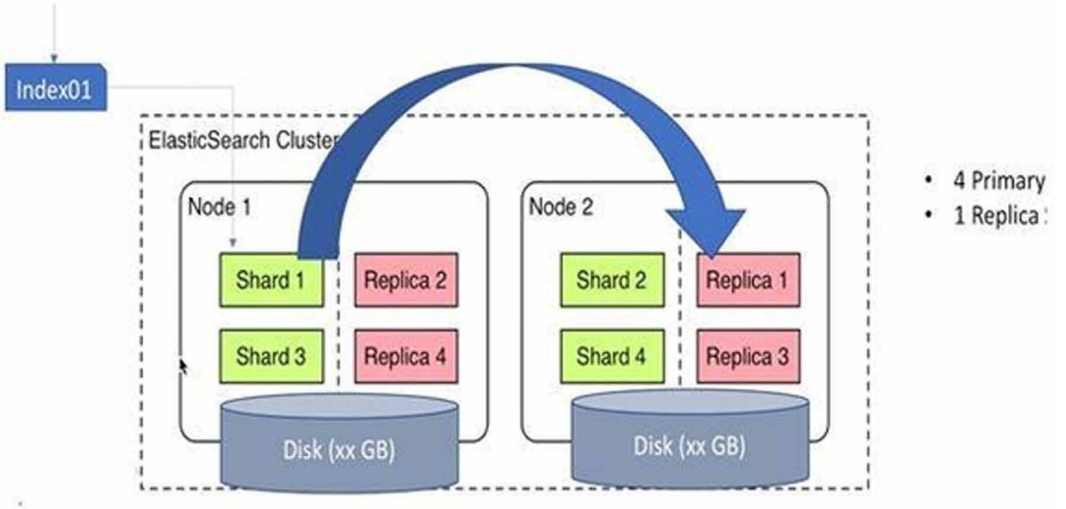
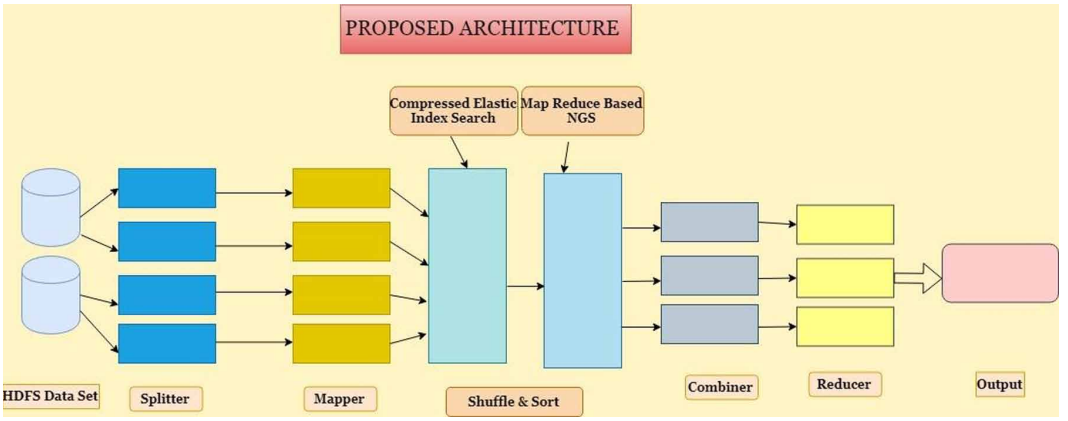


Figure 4. Architecture of CESI -MRBNGSA



minimized Metadata can be aligned in the proper way then will give it to sorting. Because of the arrangement in this NGS method, there is no time delay in creating a mapper output file. Finally, the data will give it to the Combiner part and processed by reducer for getting compressed output then it will store into HDFS storage cluster. The following Figure 4 will explain the architecture of CESI and MRBNGSA.

HDFS data set has to be taken from the Hadoop Ecosystem and Map Reduce was successfully programmed using any one of the latest languages like JAVA, PYTHON. The total architecture was designed to give output at a certain time period because of decreasing delay and increasing throughput. Map Reduce components are Name node, Data Node, Job Tracker and Task tracker.

4.1 HDFS Dataset

The proposed architecture deals with datasets of unstructured digital data from different areas like healthcare, Business, industry employee data and banking. The input has to be taken as a JSON documents generated by Mongo DB or Cassandra. The size of the input file varies depends on the data set. Since the batch processing helps to send multiple data as a JSON file with different sizes at a time.

4.2 Splitter

The splitter phase in Map Reduce helps to demolish the entire file in to different file formats of different sizes. Number of tasks has to be calculated based on the input size of the files. The input file is stored in blocks with minimum of 64MB in Hadoop 1.X, and 128 MB in Hadoop 2.X.

4.3 Mapper

The mapper part contains keys like K, V for the intermediate records. It has to map to zero or many records for MapReduce work. It has done all the work by the following formula (Wang et al., 2016),

$$NumberofMaps = Numberoftotalsizeofblocks \quad (1)$$

For example, 1GB data as a input file means in Hadoop 2.X, 128 MB blocks are created. So the No of map=1024/128=8 blocks, whereas Hadoop 1.X means No of map = 1024/64=16 blocks.

4.4 Shuffler

The shuffler part is used to give the output of the mapper to the input while processing MapReduce program. The protocol which is used to send is HTTP. The output given by the mapper () function is based on the keys allocated for all jobs.

4.5 Sort

The sort phase of Map Reduce is used to group the key based on Meta data of the input file. Again it has to compare the different keys from mapper and it is called secondary sort. Both shuffle and sort can be done simultaneously.

4.6 Combiner

Combiner is the important concept of Map Reduce paradigm and it is used to reduce the number of keys from mapper function. The key elements have decided the functionalities of combiner based on its input from the sorted phase. In partitioning of combiner phase is used to create subset of key using hash function for mapping. It can be calculated using the formula (Wang et al., 2016)

$$Numberofpartiotions = Numberofreducedtasks \tag{3}$$

4.8 Reducer

It is used as a counter to increment or decrement the values as and output and their jobs. In order to increase the count value of the reducer number of tasks completed would be zero. Otherwise the uncompleted tasks are still in the queue of reducer. The output value will be stored in a HDFS horizontal storage server or commodity server.

5. PROPOSED METHODOLODY

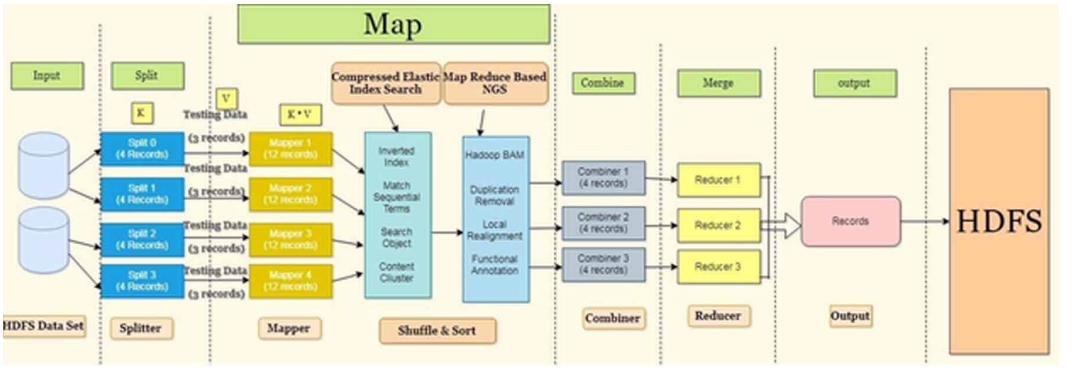
Initially, the data set which is stored in the HDFS system of a Hadoop Framework Ecosystem has been taken as an input of smaller file to this approach. Map Reduce is ready to accept the smaller files in the form of unstructured data. In the Hadoop version, 2.x takes this and sends it to the resource manager. Resource Manager has the name node as a master and data node as a slave for accessing methods. Then the job tracker of the name node assigns this job into the data node by the resource manager. Data node tasks can track by a task tracker. The problems faced by the proposed system architecture is

- Imbalanced data distribution by the clustering with in HDFS blocks
- Linear complexity of distribution of sub data sets
- Create data structure like Compressed Elastic Search Index for better performance of parallel computing
- Content Clustering and Sub data set computing

5.1 Design of CESI and MRBNGSA System

The next phase of map-reduce is splitter which will splitter the entire job assigned by RM into the number of pieces or splits. For example, input smaller files can be scattered into 4 records and 3 records are taken as training/test records. It can be denoted as K. Test records are called as V. Mapper part is responsible for assigning keys to the jobs for map () function. Once the map () function got over by sharing of keys from splitter and testing records it will be multiplied by K * V for creating the output of the mapper. Totally 12 records will be created and send it to the next phase of MR called shuffle and sorting. The following Figure 5 will explain the working principle of CESI and MRBNGSA.

Figure 5. working principle of CESIMRBNGSA



Collecting and analyzing of data is important for business intelligent security system. In HDFS number of files has been stored and it contains lot of fields such as size, log time, content, id, source, destination etc. To discover the Meta data of the input files again these files are filtered for individual analysis. For this the entire values of sets are divided in to number of subsets for making it as a content cluster based on the originality or its Meta data. The sub data set S(t) related to a specific job or task (Wang et al., 2016) could be represented as follows

$$S\left(t\right)=\left\{related\left(r,t\right),\ r\in R\right\} \tag{4}$$

where R is the collection of tasks in the mapper function. Equation 3 represents the sub dataset is equal to the number of specific task assigned by name node or resource manager. Here r denotes the related task from the given task.

5.2 Mapper-Shuffle Imbalanced Data Distribution

Assume a number of tasks from splitter have to come to launched on m-node clusters to analyze a specific sub data set S, which is distributed among n block files. Due to content clustering, most of the blocks are getting different amount of data from an each sub data set. The amount of data contained in each block is X, follow a Gamma distribution *X~ ┌ (k, w),* and assume that the all blocks arte independent. To obtain the amount of workload processed on a cluster node Z is denoted as in Z~ ┌ (nk/m, w). N is node, w is selected task and k number of times accessed by the resource manager. The total density of workload distribution(Wang et al., 2016) is based on the function,

$$f\left(Z;\frac{nk}{m},w\right)=\frac{1}{\frac{nk}{m}*\frac{wnk}{m}}\times Z\left(\left(\frac{nk}{m}\right)-1\right)\times t-\frac{Z}{\varphi} \tag{4}$$

In this equation Gamma distribution helps to distribute the tasks to the nodes which were taken from the dataset. There are initially 12 records can processed into the data processing step. Metadata of the input file has to take for experiment and it will store in the inverted index as tokens, ids. From this index value, it has to be compressed with the help of avoiding repetitive and duplicated values from the same word or sentence. On that, sequential terms of the letter occurrences have found using elastic search. The probability of workload can be calculated for cluster node (Wang et al., 2016) as follows.

$$P\left(Z < w\right) = \int_0^w f\left(x\right) tdt \tag{5}$$

The same will be calculated for workload greater than the node also.

$$P\left(Z > w\right) = \int_0^w f\left(Z\right) tdt \tag{6}$$

In general the probability of the workload size is increased in cluster node automatically m is also increases. For example given the value of k=1.5,w=9, n=1024 the probability is approximately 0 to 0.35 when we increase nodes in a cluster. All w values indicate the larger cluster nodes will result a higher chance of an imbalanced workload. When w value implies ½,1/3,1/4,0,1,2,3….n the results shows larger workloads give longer execution time for completion of the task whereas shorter workload gives low execution time to finish it off.

Number of workload is greater than the nodes then the probability value is 0 to 0.15.Imbalanced workload distribution has done with the help of above Gamma Distribution among the nodes as per the input files coming from the HDFS data set. So based on the above formulas and values if the number of nodes is increased the time taken to complete the job is very low whereas the number of workload is increased but nodes are less, eventually time has to be high for job completion.

5.3 Distribution of Sub Dataset

In general data sets can be written with the help of writing subsets of the given data set from the mapper. For example X = {1,2,3} it will be derived in to Y = {{1},{2},{3},{1,2},{1,3},{2,3},{2,1},{3,1},{3,2}} of sub data set. From this Z = {{2, 1}, {3, 1}, {3, 2}} got removed because it is repetition or duplication of data set. Now the sequential terms are matching can be calculated and it will help to find out the object. Here object denotes the files/records which are finding out by the compressed elastic search. Finally, all objects will get together as a cluster based on the same content/keys. This is called a Content cluster for further processed. The following Figure 6 gives the structure of the content cluster and elastic search index.

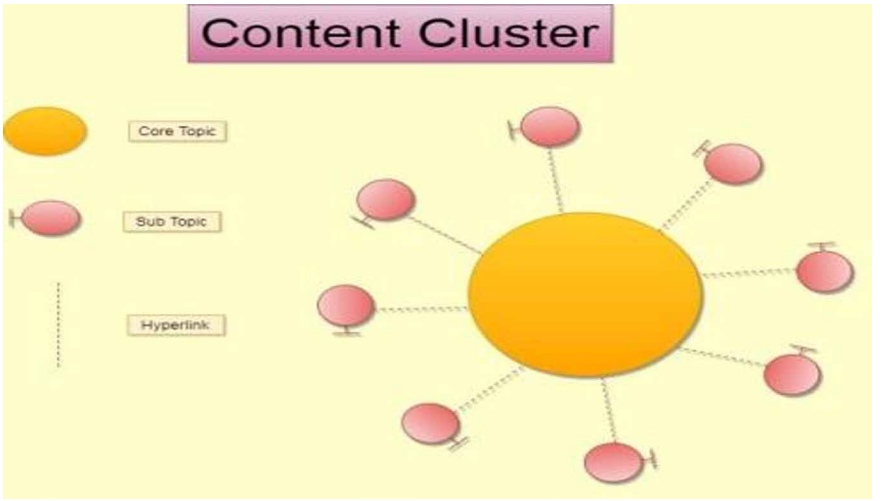
Here core topic is taken and relevant documents are considered as sub-topics which are connected through the hyperlink. For example, a hobby is the core topic then music, games; reading, cooking all is working as a sub-topic which is connected by hyperlinks with lakhs of documents. There are several content small clusters that have to be created based on the mapper output small files.

5.4 Create a Data Structure like Compressed Elastic Map

The following equation will help to create the sub dataset from the given dataset as an output from the splitter. The size of data related to each data subset over block files, that is $|b_i \cap s_j|$, i=1…..n, j=1……m where bi is the set of records on the i_{th} block and S_j is the one sub data set contained by b_i .A hash map is used to maintain this values for storing with the name of <id, quantity>. Let n be the number of block files in a data set and elastic map array information over n blocks. That array has n pointers, it is pointing over Meta data of a subset over a block file. Then the structure of Elastic Map can be in the following Figure 7

In order to maintain the size information of sub dataset over a block file b_i,the intervals are denoted as buckets and distributed to $|b_i \cap s_j|$, i=1…..n,j=1……m via one scan of the block. S_j=0 when initiating, after that it will get increase by one by one or complex values based on the buckets. Because of the content clustering, the buckets corresponding to lager data size will contain a smaller number of data sets. Non uniform buckets also occurred as instances at regular intervals.

Figure 6. Structure of Content Cluster



5.5 Content Clustering and Sub Dataset Computing

For example if we take one instance of that buckets based on the Fibonacci series, it can be (0, 1kb), (1kb,2kb),……..(64kb,n). If this method of data set will be given to the cluster nodes with block file of 64MB,with upper-bound size of 32kb,then 64M/32Kb=2048 sub data sets will be created in the buckets. This will be put it in to elastic maps with 16kb memory cost of upper bound and 1 kb of lower bound. Upper bound denotes the details of the files whereas lower bound denotes the Metadata of the file. If the file contents or less than the 1kb metadata file it will be eliminated from the datasets and realigned or reassigned to any other node. The time complexity of a sub dataset sorting before elastic map introduced is *0 (m.logm) .After elastic maps* insisted to this method their time complexity will be *0(m).* Then if m is the number of sub datasets deals with n blocks, the time complexity is *0(m*n).* To balance the workload among content cluster nodes, the total size(Wang et al., 2016) of a sub-dataset as follows

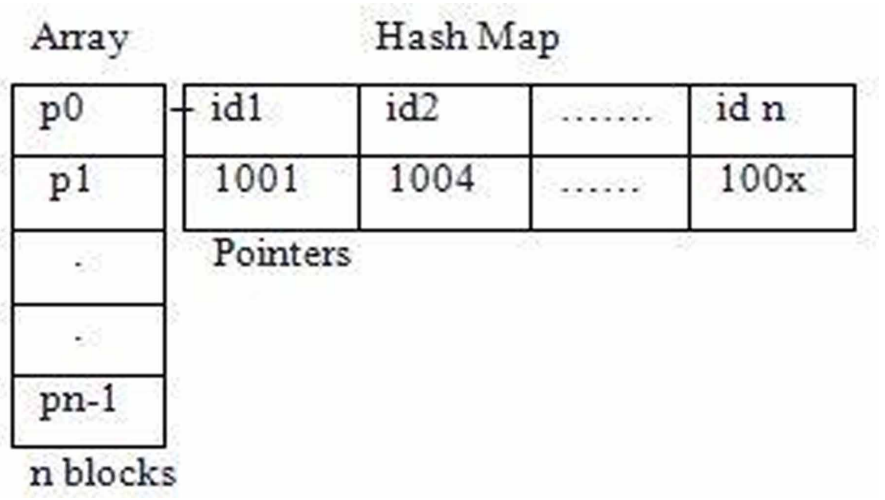
$$Z = \sum_{bj \in t_1} \left| S \cap b_j \right| + \in \times |t_1| \tag{7}$$

Where s is a given subset, t1 is the set of blocks and £ is the minimum meta data file which is less than 1 kb from the given data set. Then compressed elastic search index algorithm is proposed with new features to remove the duplicates and unwanted files which size of the input is less than 1kb.

Algorithm 1: Compressed Elastic Search Index for balanced Computing over a sub dataset

1: Let $T=\{t_0, t_1 \ldots .. t_{n-1}\}$ be the set of task to the n blocks
2: Let $|b_i \cap s_j|$ be the size of sub dataset an in block j.
3: Let w be the workload on cluster node.
Steps:
1: Compute the average workload W with Z where m is total number of cluster nodes
2: While $|T_i| \neq 0$ do
3: if a work is selected as a balanced task then
4: if $| s_j | \neq 0$ then

Figure 7. Elastic Map Structure




```
5: Find bi € s such that
6: Assign t1 to the nest node
7: else
8: Find sj € n such that
9: Assign txto the next node
10: endif
11: Remove tx from T
12: For all T={t0,t1…..tn-1} in T do
13: end for
14: end if
15: end while
```

After the results came from the above algorithm, that will be stored in HDFS block files for sending it to the sorting level. Elastic Map is used to minimize the data r\transferred with sub dataset distributions. Further it can be optimized for the better solutions.

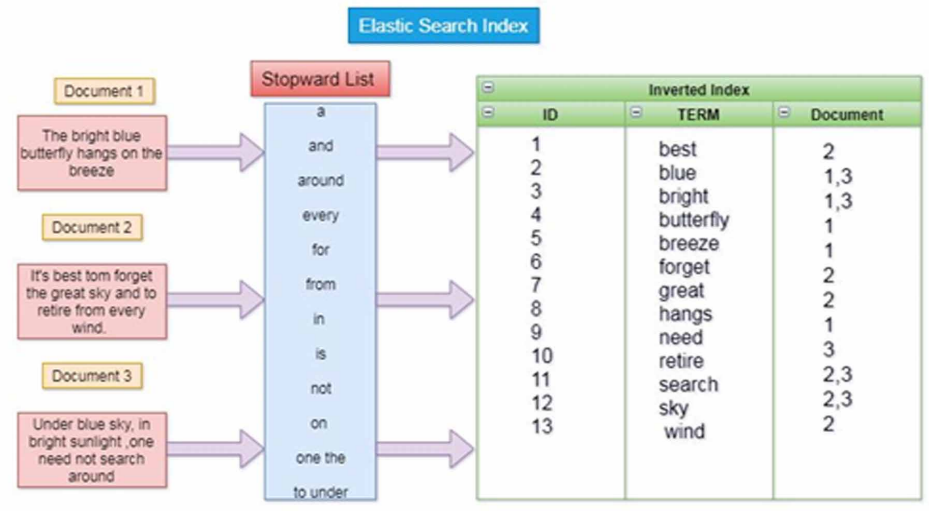
Elastic Search Index gives the optimized document words from a collection of documents. Initially, it will remove the entire stop ward list for creating Metadata. Again those data documents are checking for duplication and repetition using seed and extend algorithm. There the documents will be segregated as ID, TERM and Document Name for creating sets. For the subset, this data set will remove the repeated occurrence letters in the sequence of words. Finally, one document will give the exact result of the search word given by the client in the mapper section of the Map Reduce phase. The next step is to remove duplicated data removal by algorithm and the original data has to be realigned for processing. The following Figure 8 gives the elastic search index working principle.

Finally, the sorting functions are using annotations to get the final records from the map part and give input to the combiner section. These combiner 4 records will get reduce into a single file converted record and given as an output to the HDFS files system. While getting output from the reducer part, if we calculate the latency and throughput it slightly decreased from the existing models.

5.6 Applying Map Reduced Based NGS Approach

The important phase of the map reduces after a content cluster is to sort the collected input data in a cluster. For this, we have used Map-Reduce Based Next Generation Sequencing Approach (MRBNGSA) to do the alignment of files ordered by keys. To perform alignment of incoming smaller

Figure 8. Elastic Search Index working principle



files we are using java based Hadoop BAM library files for distributed processing of data in Hadoop. It accepts the scalability of aligned reads in the Hadoop Distributed Computing Framework. It works as an integration layer between analysis and file applications as Binary Alignment Map files. This issue will be solved by this Hadoop BAM by providing algorithms for implementing sorting functions in the map to reduce concepts clearly.

5.7 Partition of Tasks and Placement

The shuffle phase given the output of Metadata files driven by compressed elastic search index their matches of ids and quantity <id,quantity> will be mapped. For this a new approach and algorithm called map reduce based next generation scheduling is developed and working for various datasets for optimization. For those activities initial step of partitioning the metadata files has to be done with mapper outputs. Let us considered given m map output partitions with sizes of q1,q2,q3,…..qm, find the placements on x nodes, G1,G2…Gn that minimize (Wang et al., 2016) the placement difference in

$$F = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p - \sum_{j \in G_i} q_i} \tag{8}$$

Where F is the average data size on one node.

```
Algorithm 2. Partition and Placement of Jobs
1: variables: list of partitions q,list of nodes x,size of
partitions g
2: assign the number of nodes, partitions and their size
3: sort list q in descending order of partition sizes
4: for i←1 to m do
5: min_node← G[i]
6: for j←1 to x do
```

```
7: if G[j].size < min_node.size then
8: min_node←G[j]
9: end if
10: end for
11: min_node.place(q[i])
12: end for
```

The above algorithm is used to partitioning the tasks and assigns it in to number of nodes. Based on the algorithm the tasks can be assigned to the nearest nodes and it can be accessed based on the size of tasks. This algorithm is based on two things. 1. Largest partition is picking first and assign for the node 2. Less workload selected first and picking their destinations. It repeated until all the partitions are assigned.

Based on the above two approaches this MRBNGS algorithm partitioning the entire jobs entered in to the sorting phase. For example if we take 1 TB of data as a input file,64MB and 128MB size blocks have created respectively according to the Hadoop versions for map reduce functions.. There are three positions namely 1,2,3 will be created by partitioning algorithm due to size of blocks .In position 1, first 100^{th} position of block data has taken for the consideration for metadata file creation. In that duplication and repetition files will be deleted and reduced the entire file system in to 64/128MB blocks. In position 2, 101^{th} position to 700^{th} position data blocks will be considered for optimization. In position 3, remaining all positions (800 to 1024) will be considered to do the same partitioning and positioning of all jobs. In this approach minimum of 1KB (1024) positions to maximum of 1GB positions will be considered based on the input file size. This is repeats the iterative works up to the maximum file size continuously. The following Figure 9 will give the structure of MRBNGS clearly.

The positions are stored and running in a queue based method along with its priority. Any remote systems will enter in to the cluster for up gradation that will be automatically reassigned the id, quality of input files because of the Ethernet environment given in a network system. It will reduce the delay of the partitioning and sorting phase of Map Reduce and working with different test beds like Hadoop A, DynMR, and Sailfish system. Totally proactive placement of map out partitions, this approach will not give any delay when compared with previous systems. There are other methods like globally sorted partitions and overlapping in reduce phase is used in this approach. When comparison takes place in the total system, overall delay due to file size and workload will be reduced based on the new algorithm and approaches.

6. EXPERIMENTAL RESULTS

The experimental setup of the model has been created as a cluster of commodity computers and Hadoop Distributed File System (HDFS) has used for repository data. Thus, Hadoop 2.51 version has been installed in this system setup and Ubuntu Linux with kernel 2.6.24 operating system is in every computer of the cluster. Since it is a multi-node cluster set up one of the systems is configured as the master node with 16 GB of RAM and others are configured as slave nodes with 16/4 GB RAM. Master node processor clock speed at 2.4 GHZ with 4 core virtualized CPU. This configuration made very simple with low-cost components and the number of nodes also limited for checking the speed and time taken off the map-reduce programming work. The following Table 4 narrates the Hadoop Cluster Configurations details and lab setup.

7. RESULT AND DISCUSSIONS

The experimental results have been taken from modern tools like Hadoop BAM and GATK. The factors considered for results is execution time, size of data, workload distribution, number of data sets, and minimal/maximum average time taken to complete the process. The unit of all timing is based on RPKM (Reads per kilo base million mapped reads). The following figures will explain the

Figure 9. Structures of MRBNGS

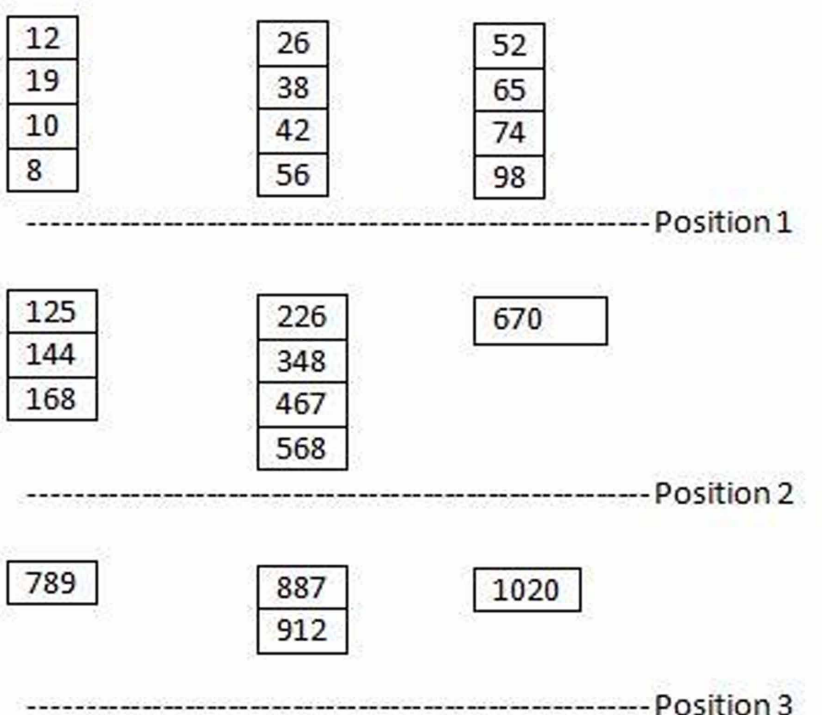


Table 4. Hadoop Cluster Configuration and setup

S No	Machine	Description	Memory(GB) RAM	Disks
1	Master	Hadoop 2.51, Ubuntu Linux with kernel 2.6.24 2.4 GHz, 4 core Intel Xeon E5530	16	10
2	Node1	Hadoop 2.51, Ubuntu Linux with kernel 2.6.24 2.4 GHz, 4 core Intel Xeon E5530	8	10
3	Node 2	Hadoop 2.51, Ubuntu Linux with kernel 2.6.24 2.4 GHz, 4 core Intel Xeon E5530	8	10
4	Node3	Hadoop 2.51, Ubuntu Linux with kernel 2.6.24 2.4 GHz, 4 core Intel Xeon E5530	8	10
5	Node4	Hadoop 2.51, Ubuntu Linux with kernel 2.6.24 2.4 GHz, 4 core Intel Xeon E5530	8	8
6	Node 5	Hadoop 2.51, Ubuntu Linux with kernel 2.6.24 2.4 GHz, 4 core Intel Xeon E5530	4	9

results taken from the experimental model and indicate our model is slightly gaining the features of other models and time is to be reduced. The following Figure 10, 11, 12 and13 indicate the results of our new approach running on the Master-Slave Multi-node Hadoop cluster setup. The factors taken for the experiments are the mapping of data, sequence match, duplication removal, and realignment, number of tasks, speedup, short read length and number of short reads. Figure 10 indicates the mapping

Figure 10. Speed up time completion of multiple jobs

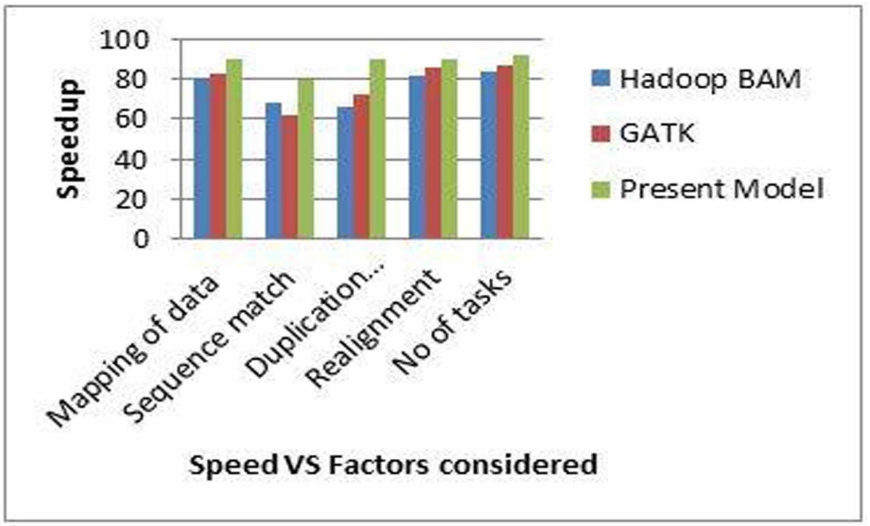
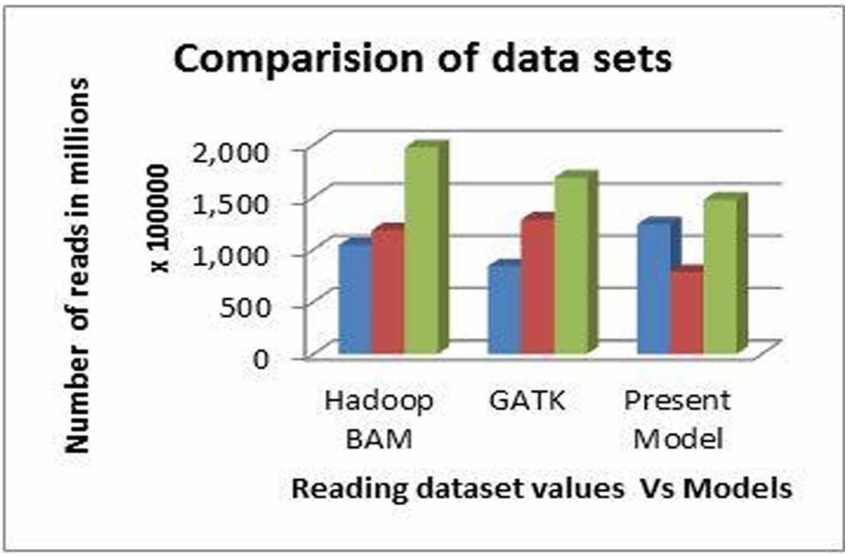


Figure 11. Number of short reads from various dataset



of data, sequence match, duplication removal, and realignment, the number of tasks accessed in the Hadoop BAM and the GATK system model and it also compared with our new approach.

It seems that there is a slight change in task completion time that is speed up data that is accessed. It clearly shows our new model increases a minimum of 5% to a maximum of 10% time completion level.

Figure 11 indicates the number of short reads from the different data sets taken from various sources and accessed through our model, there are more (10%) short reads has taken for input file because of compressed elastic search and new alignment techniques. If data sets are large the number

Figure 12. Large file of short reads

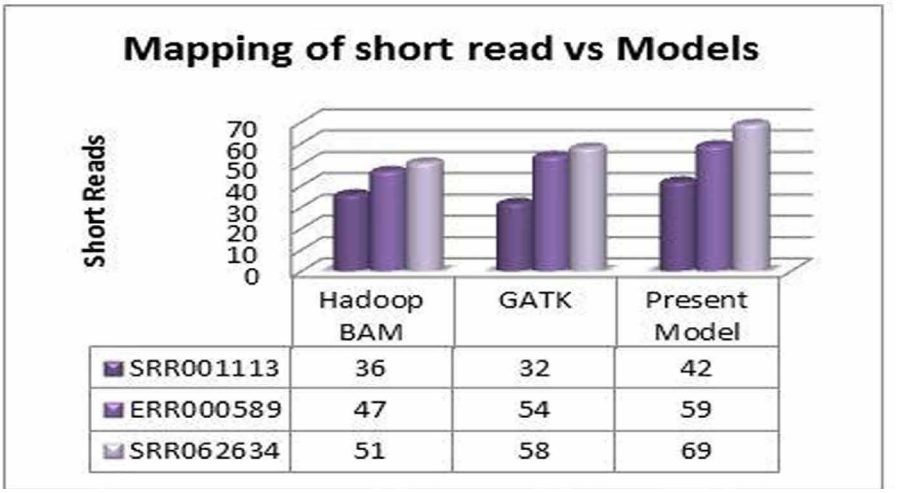
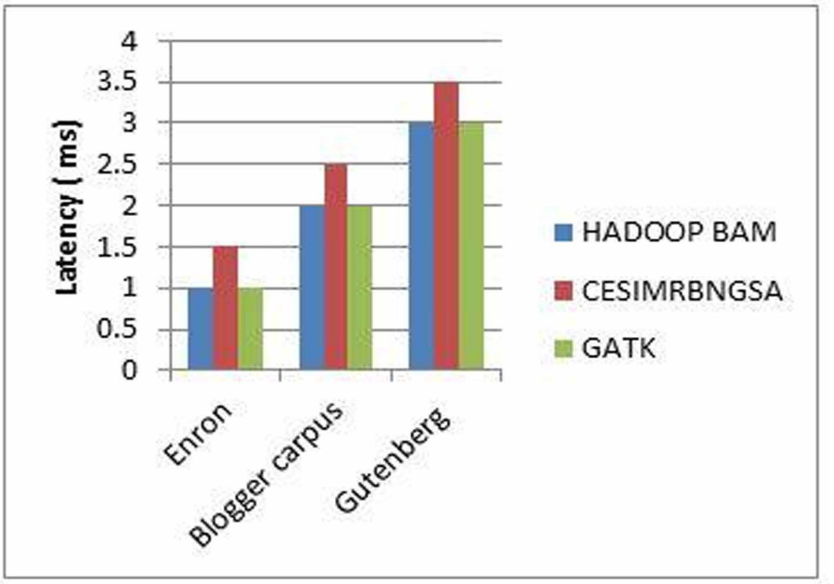


Figure 13. Throughput



of short reads using this approach is very high. The data sets taken for reading and writing values is Enron, Blogger Corpus, Gutenberg at minimum of 512 MB to maximum of 1 GB word/text files were taken. Initially it is reduced by normal map reduce program by Hadoop environment. After that the file size has reduced because of preprocessing and meta data reduction using compressed elastic search index method again it will be given as a input to the map reducer part. Due to the algorithm of elastic search again the file vales are compressed by its own data for retrieval.

Figure 12 indicates the time taken to read the total file length of shorter reads from different data set. Our model increased the length by 5% gives more efficient retrieval of data from a huge data set.

Figure 13 indicates the percentage of throughput increases by our new approach in the Hadoop Apache framework. It shows the time taken to complete all the multiple jobs assigned in Map Reduce by the clients have been reduced because of node failure and single point failure problems also job failure problems.

8. APPLICATIONS

This proposed system will help to increase the speed of data processing from huge data set clusters like social media, cloud framework for business applications (Auxilia et al., 2020), Sustainable development (Zelinka & Amadei, 2019), Climate change monitoring Naugle, A. B et al. (2019)), the health care sector (Majhi, 2018), (Dutta, 2017), Multi object tracking in Transportations (Elfouly et al., 2017), banking sector, education sector, and industries. Moreover unwanted and duplicated data can be removed from the data cleaning process because the data storage method in HDFS will be effective for ETL. This approach helps to get the exact data search of the users given in the search engine. More applications are used in health care for their huge volume the speed will be increased because of compressed elastic search, memory occupied by the map-reduce concept in the Hadoop framework is released very fast when compared with existing models. A lot of models are used to find the solutions for latency and throughput this approach gives increased 15% gain of output format unstructured data. Clusters are saved according to the formats segregated by elastic search and content cluster it will be giving results immediately based on its size. Huge data set retrieval problems can be solved through this approach but security is also a concern to the implementation part. Overall performance and improvement of this approach are to reduce the time taken of completed failure jobs because of node failure will be increased with respect to speed.

9. CONCLUSION

This approach is used to find the solution for latency and throughput issues in the map-reduce concept of the Hadoop Framework in Big data Analytics. Moreover, the data which are cleaned by the help of duplication removal and repetitive removal will give sufficient speed to retrieve data from the data warehouse. Data realignment is an innovative idea where it processed by CESI and MRBNGSA methods. This has to reduce the complexity level of extracting complex data set from a data warehouse. Big data analytics will help the users for finding statistics of the data been generated by manual or machine-related. Data visualization helps to improve this approach for the development of cleaning and processing algorithms. Hadoop BAM and GATK tools are used to find the time taken to complete the process based on RPKM (Reads per kilo base million mapped reads). It gives the ratio between the mapper reads and reduced outputs based on the results given by modern tools. Thus improving data processing speed in big data analytics using HDFS method by Compressed Elastic Search Index (CESI) with Map Reduce- Based Next Generation Sequencing Approach (MPBNGSA) gives an improvement in this model further will be in a large scale using high-end machines has to be developed.

10. FUTURE ENHANCEMENT

Though the results and discussions intimate the speed of the data has increased by using CESI and MRBNGSA methods in big data analytics, HDFS storage can be executed for certain data set and modern tools. Future experiments will be working on HPC clusters and Drep clusters based on the Apache framework and find out the time taken to complete the jobs. Moreover, Apache SPARK tool also used to get faster results than this system the cost of that approach is very high due to the RAM cost. Eventually, huge data sets and various data sets from different sectors will be processed by this approach in the future in order to develop the new approach or algorithm inevitably. Most of the

failure jobs from Map Reduce occurred because of node failures and cluster selection. Future work will be on cluster forming with a lot of Machine Learning and Artificial Intelligence concepts with modern tools in Big Data Analytics. Next decade Big Data Analytics will create more impact on all areas in this real world, generated data storage will be managed by both software and hardware but data retrieval and processing will create big issues on complex data sets.

REFERENCES

Alfonseca, E., Garrido, G., Delort, J.-Y., & Peñas, A. (2013). WHAD: Wikipedia historical attributes data. *Language Resources and Evaluation*, *47*(4), 1163–1190. doi:10.1007/s10579-013-9232-5

Auxilia, M., Raja, K., & Kannan, K. (2020). Cloud-Based Access Control Framework for Effective Role Provisioning in Business Application. *International Journal of System Dynamics Applications*, *9*(1), 63–80. doi:10.4018/IJSDA.2020010104

Duggal, P. S., & Paul, S. (2013). Big Data analysis: Challenges and solutions. *International Conference on Cloud, Big Data and Trust*, *15*, 269–276.

Dutta, P. (2017). Decision making in medical diagnosis via distance measures on interval valued fuzzy sets. *International Journal of System Dynamics Applications*, *6*(4), 63–83. doi:10.4018/IJSDA.2017100104

Elfouly, F. H., Ramadan, R. A., Mahmoud, M. I., & Dessouky, M. I. (2017). Efficient data reporting in a multi-object tracking using WSNs. *International Journal of System Dynamics Applications*, *6*(1), 38–57. doi:10.4018/IJSDA.2017010103

Guo, Y., Rao, J., Cheng, D., & Zhou, X. (2016). ishuffle: Improving hadoop performance with shuffle-on-write. *IEEE Transactions on Parallel and Distributed Systems*, *28*(6), 1649–1662. doi:10.1109/TPDS.2016.2587645

Khan, N., Alsaqer, M., Shah, H., Badsha, G., Abbasi, A. A., & Salehian, S. (2018). The 10 Vs, issues and challenges of big data. *Proceedings of the 2018 International Conference on Big Data and Education*, 52–56. doi:10.1145/3206157.3206166

Li, J., Wang, J., Lyu, B., Wu, J., & Yang, X. (2018). An improved algorithm for optimizing MapReduce based on locality and overlapping. *Tsinghua Science and Technology*, *23*(6), 744–753. doi:10.26599/TST.2018.9010115

Luna, J. M., Cano, A., Pechenizkiy, M., & Ventura, S. (2016). Speeding-up association rule mining with inverted index compression. *IEEE Transactions on Cybernetics*, *46*(12), 3059–3072. doi:10.1109/TCYB.2015.2496175 PMID:26800557

Majhi, S. K. (2018). An efficient feed foreword network model with sine cosine algorithm for breast cancer classification. *International Journal of System Dynamics Applications*, *7*(2), 1–14. doi:10.4018/IJSDA.2018040101

McCreadie, R., Macdonald, C., & Ounis, I. (2012). MapReduce indexing strategies: Studying scalability and efficiency. *Information Processing & Management*, *48*(5), 873–888. doi:10.1016/j.ipm.2010.12.003

Mondal, S., & Khatua, S. (2019). Accelerating pairwise sequence alignment algorithm by mapreduce technique for next-generation sequencing (ngs) data analysis. In *Emerging Technologies in Data Mining and Information Security* (pp. 213–220). Springer. doi:10.1007/978-981-13-1498-8_19

Nabavinejad, S. M., Goudarzi, M., & Mozaffari, S. (2016). The memory challenge in reduce phase of mapreduce applications. *IEEE Transactions on Big Data*, *2*(4), 380–386. doi:10.1109/TBDATA.2016.2607756

Remya, S., Somula, R., Nalluri, S., Vaishali, R., & Sasikala, R. (2018). Big Data for Satellite Image Processing: Analytics, Tools, Modeling, and Challenges. In Big Data Analytics for Satellite Image Processing and Remote Sensing (pp. 133–150). IGI Global.

Rexie, J. A. M., & Raimond, K. (2019). Evolution of Methods for NGS Short Read Alignment and Analysis of the NGS Sequences for Medical Applications. In *Computer Aided Intervention and Diagnostics in Clinical and Medical Images* (pp. 135–142). Springer. doi:10.1007/978-3-030-04061-1_13

Samaddar, S., Sinha, R., & De, R. K. (2018). A model for distributed processing and analyses of ngs data under map-reduce paradigm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *16*(3), 827–840. doi:10.1109/TCBB.2018.2816022 PMID:29993814

Sankar, S., Srinivasan, P., Luhach, A. K., Somula, R., & Chilamkurti, N. (2020). Energy-aware grid-based data aggregation scheme in routing protocol for agricultural internet of things. *Sustainable Computing: Informatics and Systems*, *28*, 100422. doi:10.1016/j.suscom.2020.100422

Sankar, S., Srinivasan, P., Ramasubbareddy, S., & Balamurugan, B. (2020). Energy-aware multipath routing protocol for internet of things using network coding techniques. *International Journal of Grid and Utility Computing*, *11*(6), 838–846. doi:10.1504/IJGUC.2020.110899

Sennan, S., Balasubramaniyam, S., Luhach, A. K., Ramasubbareddy, S., Chilamkurti, N., & Nam, Y. (2019). Energy and Delay Aware Data Aggregation in Routing Protocol for Internet of Things. *Sensors (Basel)*, *19*(24), 5486. doi:10.3390/s19245486 PMID:31842437

Sennan, S., Ramasubbareddy, S., Luhach, A. K., Nayyar, A., & Qureshi, B. (2020). CT-RPL: Cluster Tree Based Routing Protocol to Maximize the Lifetime of Internet of Things. *Sensors (Basel)*, *20*(20), 5858. doi:10.3390/s20205858 PMID:33081218

Sennan, S., Somula, R., Luhach, A. K., Deverajan, G. G., Alnumay, W., Jhanjhi, N. Z., Ghosh, U., & Sharma, P. (2020). Energy efficient optimal parent selection based routing protocol for Internet of Things using firefly optimization algorithm. *Transactions on Emerging Telecommunications Technologies*, 4171.

Somula, R., Anilkumar, C., Venkatesh, B., Karrothu, A., Kumar, C. S. P., & Sasikala, R. (2019). Cloudlet Services for Healthcare Applications in Mobile Cloud Computing. *Proceedings of the 2nd International Conference on Data Engineering and Communication Technology*, 535–543. doi:10.1007/978-981-13-1610-4_54

Somula, R., & Sasikala, R. (2018). Round Robin with Load Degree: An Algorithm for Optimal Cloudlet Discovery in Mobile Cloud Computing. *Scalable Computing: Practice and Experience*, *19*(1), 39–52. doi:10.12694/scpe.v19i1.1392

Somula, R., & Sasikala, R. (2019). A Load and Distance Aware Cloudlet Selection Strategy in Multi-Cloudlet Environment. *International Journal of Grid and High Performance Computing*, *11*(2), 85–102. doi:10.4018/IJGHPC.2019040105

Somula, R. S., & Sasikala, R. (2018). A Survey on Mobile Cloud Computing: Mobile Computing+ Cloud Computing (MCC= MC+ CC). *Scalable Computing: Practice and Experience*, *19*(4), 309–337. doi:10.12694/scpe.v19i4.1411

Sundarakumar, M. R., & Mahadevan, G. (n.d.). *Improving Speed and Accuracy of Image Retrieval using Elastic Search and Features Nearest Neighbor Search*. Academic Press.

Velusamy, K., Venkitaramanan, D., Vijayaraju, N., Suresh, G., & Madhu, D. (2013). Inverted indexing in big data using hadoop multiple node cluster. *International Journal of Advanced Computer Science and Applications*, *4*(11). Advance online publication. doi:10.14569/IJACSA.2013.041122

Wang, J., Zhang, X., Yin, J., Wang, R., Wu, H., & Han, D. (2016). Speed up big data analytics by unveiling the storage distribution of sub-datasets. *IEEE Transactions on Big Data*, *4*(2), 231–244. doi:10.1109/TBDATA.2016.2632744

Wani, M. A., & Jabin, S. (2018). Big data: issues, challenges, and techniques in business intelligence. In *Big data analytics* (pp. 613–628). Springer. doi:10.1007/978-981-10-6620-7_59

Weets, J.-F., Kakhani, M. K., & Kumar, A. (2015). Limitations and challenges of HDFS and MapReduce. *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 545–549. doi:10.1109/ICGCIoT.2015.7380524

Xia, D., Li, H., Wang, B., Li, Y., & Zhang, Z. (2016). A map reduce-based nearest neighbor approach for big-data-driven traffic flow prediction. *IEEE Access: Practical Innovations, Open Solutions*, *4*, 2920–2934. doi:10.1109/ACCESS.2016.2570021

Yang, A., Zhu, S., Li, X., Yu, J., Wei, M., & Li, C. (2018). The research of policy big data retrieval and analysis based on elastic search. *2018 International Conference on Artificial Intelligence and Big Data (ICAIBD)*, 43–46. doi:10.1109/ICAIBD.2018.8396164

Yu, W., Wang, Y., Que, X., & Xu, C. (2013). Virtual shuffling for efficient data movement in mapreduce. *IEEE Transactions on Computers*, *64*(2), 556–568. doi:10.1109/TC.2013.216

Zelinka, D., & Amadei, B. (2019). Systems Approach for Modeling Interactions Among the Sustainable Development Goals Part 1: Cross-Impact Network Analysis. *International Journal of System Dynamics Applications*, *8*(1), 23–40. doi:10.4018/IJSDA.2019010102

Zhang, Q., Zhani, M. F., Yang, Y., Boutaba, R., & Wong, B. (2015). PRISM: Fine-grained resource-aware scheduling for MapReduce. *IEEE Transactions on Cloud Computing*, *3*(2), 182–194. doi:10.1109/TCC.2014.2379096

Sundara Kumar M. R. (PhD) is presently working as a research scholar in the department of computer science and engineering at AMC Engineering College, Bangalore. He has 10+ years of teaching and 5 Years of research experience. His research interests include Cloud computing, computer networks, networks management, Big Data Analytics, cryptography. He has published 4 Scopus indexed papers at various international journals He has presented 06 papers in international conferences, 04 papers in national conferences. He has organized few workshops, seminar, guest lecturers held at various college levels. He has delivered seven technical talks at different engineering colleges with the theme of Cloud computing issues and challenges, Cloud security concepts.

Bharat S. Rawal is an Associate with the Department of Cyber Security at Gannon University, PA, USA. He has more than 50 research publications in different journals and conferences. His research interest are Cybersecurity, Big Data, IoT, HPC, and Cloud.