IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 7, Issue 3, March 2020

A Survey on Job Schedulers of MapReduce

Vijayalakshmi P[1], Vinutha D C[2], G T Raju[3]

Student,Department of Information Science & Engineering,Vidyavardhaka college of Engineering,Visvesvaraya Technological University, Karnataka[1]

Associate Professor, Department of CSE, Vidyavardhaka College of Engineering, Mysuru, Visvesvaraya Technological University, Karnataka[2]

Professor, Department of CSE, RNS Institute of Technology,Bengaluru Visvesvaraya Technological University, Karnataka[3]

Abstract: One of the fastest growing applications is big data. Analysis of big data is must to gain higher productivity and efficient data utilization. Map Reduce is emerging as an important programming model for large-scale data-parallel applications such as web indexing, data mining, and scientific simulation. Hadoop is an open source implementation of the Map Reduce framework. It is a highly robust system model in the form of scalable and fault tolerant distributed system and it is important for data storage and its processing. Goal of Hadoop is to offer efficient and high performance processing of big data application. Hadoop clusters composed of hundreds of nodes to process terabytes of user data. This paper gives a survey of different job schedulers of Map Reduce.

Keywords: Cloud Computing, HDFS, Map Reduce, Hadoop.

1. INTRODUCTION

Cloud computing is a technology where the data resources are shared rather than owing the personal server applications or devices. Cloud computing is in general stated as an internet based computing/service where all the comprising functions such as data storage, servers and applications are facilitated to certain organizations using Internet connectivity. Cloud has become an inevitable need for majority of IT organizations. Cloud applications such as data storage, data retrieval and data portability have become significant requirements for IT and ITeS organizations dealing with cloud computing and Big Data. With the high paced increase in population using smart applications, the emerging vast amount of data causes the applications to become data-insensitive in behavior. The predominant techniques for data insensitive applications are various search engines, online retail operations, social media, web mails etc. All these data-insensitive applications are based on data mining and related indexing approach that require the spread out of data sizing from few gigabyte to multiple Petabytes. Thus, considering such huge data collection and its efficient retrieval, a concept called Big Data came into existence.

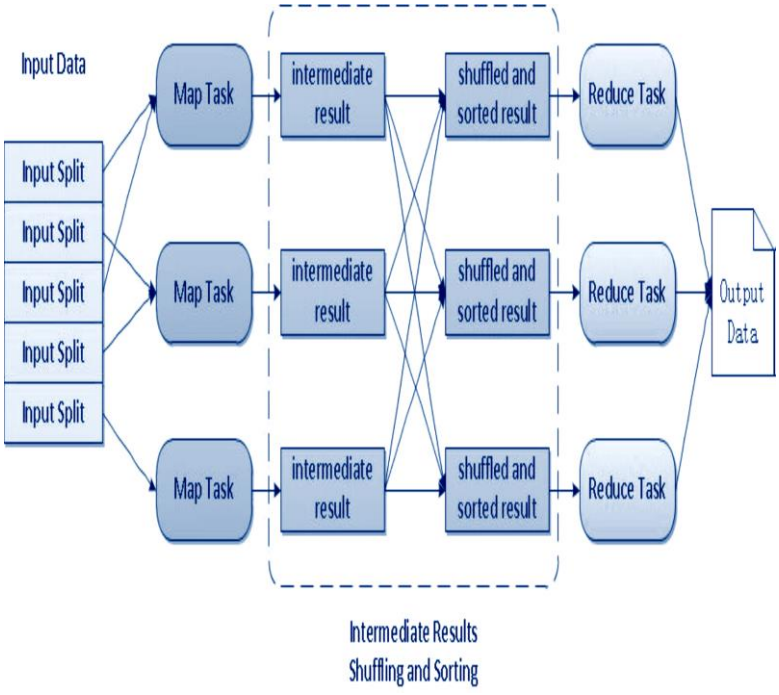

Figure 1. The workflow of Map Reduce framework

DOI 10.17148/IARJSET.2020.7318

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 7, Issue 3, March 2020

Apache Hadoop [2] is a framework for distributed applications. Doug Cutting is inspired by Google Map Reduce and he introduced Hadoop Map Reduce for distributed applications. Apache Hadoop mainly deals with Big Data processing on distributed environment. Hadoop distributed file system (HDFS) is the storage area in this framework. Map Reduce is the technique for Big Data processing and analyzing in parallel

2. BACKGROUND: HADOOP AND MAP REDUCE

2.1. Map Reduce overview

Map Reduce [1] is a framework pioneered by Google for processing large amounts of data in a distributed environment. Map Reduce adopts a divide-and-conquer approach for data-intensive applications [14]. In Map Reduce model, the execution of an application can be divided in two phases: map and reduce, as illustrated in Figure 1. In the map phase, amounts of map tasks process data blocks independently. After all map tasks are finished, the reduce phase begins. The intermediate results of map tasks are shuffled, sorted and then processed in parallel by one or more reduce tasks.

2.2 Hadoop

Hadoop [2, 21] is the open source implementation of the Map Reduce framework. Due to the simplicity of its programming model and the run-time tolerance for node failures, Map Reduce is widely used by companies such as Facebook, the New York Times, etc. Furthermore, scientists also employ Hadoop to acquire scalable and reliable analysis and storage services. A Hadoop cluster is composed of two parts: *Hadoop Distributed File System and Map Reduce.*

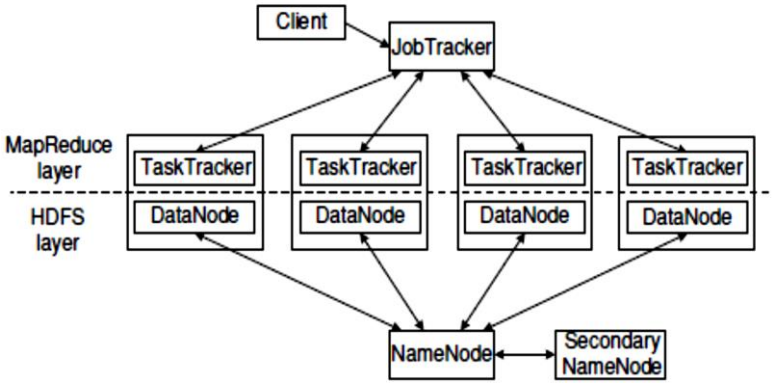

Figure 2: Hadoop Architecture

Fig 2 shows the Hadoop Architecture [3, 21]. A Hadoop cluster uses Hadoop Distributed File System (HDFS) to manage its data. HDFS provides storage for the Map Reduce job's input and output data. It is designed as a highly fault-tolerant, high throughput, and high capacity distributed file system. It is suitable for storing terabytes or petabytes of data on clusters and has flexible hardware requirements, which are typically comprised of commodity hardware like personal computers. The significant differences between HDFS and other distributed file systems are: HDFS's write once-read-many and streaming access models that make HDFS efficient in distributing and processing data, reliably storing large amounts of data, and robustly incorporating heterogeneous hardware and operating system environments. It divides each file into small fixed-size blocks (e.g., 64 MB) and stores multiple (default is three) copies of each block on cluster node disks. The distribution of data blocks increases throughput and fault tolerance. Fig 3 shows the Hadoop Distributed File System Architecture which follows the master/slave architecture. The master node is called the Name node which manages the file system namespace and regulates client accesses to the data. There are a number of worker nodes, called Data nodes, which store actual data in units of blocks. The Name node maintains a mapping table which maps data blocks to Data nodes in order to process write and read requests from HDFS clients. It is also in charge of file system namespace operations such as closing, renaming, and opening files and Directories. Hadoop Distributed File System [3] allows a secondary Name node to periodically save a copy of the metadata stored on the Name node in case of Name node failure. The Data node stores the data blocks in its local disk and executes instructions like data replacement, creation, deletion, and replication from the Name node. A Data node periodically reports its status through a heartbeat message and asks the Name node for instructions. Every Data node listens to the network so that other Data nodes and users can request read and write operations. The heartbeat can also help the Name node to detect connectivity with its Data node. If the Name node does not receive a heartbeat from a Data node in the configured period of time, it marks the node down. Data blocks stored on this node will be considered lost and the Name node will automatically replicate those blocks of this lost node onto some other Data nodes. A Hadoop cluster uses slave (worker) nodes to execute map and reduce tasks.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 7, Issue 3, March 2020

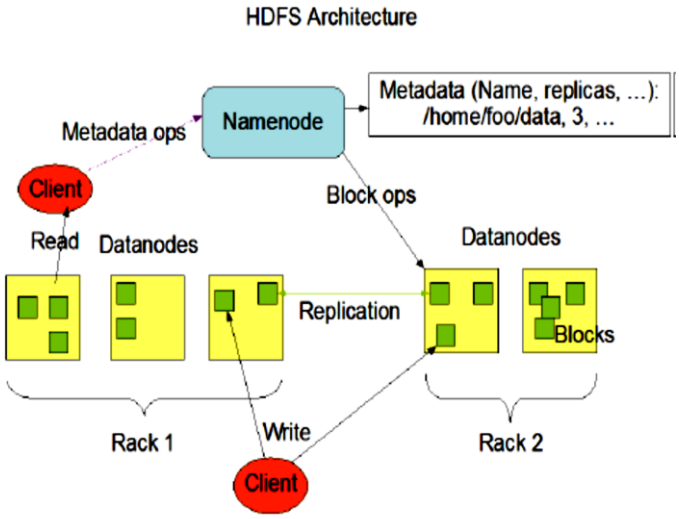

Figure 3: HDFS Architecture

3. SCHEDULERS OF MAP REDUCE

3.1. LATE: Longest Approximate Time to End.

M. Zaharia et al. [4] designed a new scheduling algorithm for speculative execution, the goal of speculative execution is to minimize a job's response time. Response time is most important for short jobs where a user wants an answer quickly, such as queries on log data for debugging, monitoring and business intelligence. It is highly robust to heterogeneity and it is based on three principles: i) prioritizing tasks to speculate, ii) selecting fast nodes to run on iii) capping speculative tasks to prevent thrashing. When a node has an empty task slot, Hadoop chooses a task for it from one of three categories. First, any failed tasks are given the highest priority. Second, non-running tasks are considered, specially the map tasks that have local data on this node. Third, the tasks which need to execute speculatively. LATE reduces Hadoop's response time by a factor of 2. LATE [13] scheduling algorithm takes the heterogeneity [19] assumptions into consideration, but has poor performance due to the static manner in computing the progress of the tasks and is not suitable for environments with dynamic loading. Neither Hadoop nor LATE schedulers are desirable in heterogeneous [20] environment. Zhao Li et al [15] proposed open flow Scheduler for heterogeneous clusters to reduce network traffic during execution of Map reduce jobs.

To improve the performance of Map Reduce in heterogeneous environments, Tao Gu et al [18] proposed a data prefetching mechanism which can fetch the data to corresponding compute nodes in advance. This mechanism has improved the performance of the job execution up to 15%.

3.2. SAMR [5, 13]: A Self-Adaptive Map Reduce Scheduling Algorithm

This approach Computes progress score of tasks more accurately than LATE, thus this scheduler launches backup tasks for really slow tasks that prolong job Execution time. Self-Adaptive Map Reduce scheduling algorithm (SAMR) uses historical Information to adjust stage weights of map and reduce tasks when estimating task execution times. However, SAMR does not consider the fact that for different types of jobs their map and reduce stage weights may be different. Even for the same type of jobs, different data sets may lead to different weights. M. Berekmeri et al [16] developed a dynamic model of a Map Reduce called a PI controller by adding a feed-forward controller to improve response time but the other metrics such as throughput and availability is not taken in to account. Jongse Park et al [20] proposed a dynamic VM reconfiguration technique for data-intensive computing on clouds, called Dynamic Resource Reconfiguration (DRR). Dynamic Resource Reconfiguration can adjust the computing capability of individual VMs to maximize the utilization of resources and Focuses on data locality. Dynamic Resource Reconfiguration can improve the throughput of Hadoop jobs by 15% on average with a constrained Network connection.

3.3. ESAMR (Enhanced self-adaptive Map Reduce scheduling):

Xiaoyn Sun (2012) proposed the ESAMR [6,13] algorithm to overcome LATE, SAMR problems. ESAMR classifies the historical information stored on every node into k clusters using a machine learning technique. If a running job has completed some map tasks on a node, ESAMR records the job's temporary map phase weight on the node according to the job's map tasks completed on the Node. The temporary M1 weight is used to find the cluster whose M1 weight is the closest. ESAMR then uses the cluster's stage weights to estimate the job's map tasks' TimeToEnd on the node and Identify slow tasks that need to be re-executed. If a running job has not completed any map task on a node, the average of all k clusters' stage weights are used for the job. In the reduce stage, ESAMR

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 7, Issue 3, March 2020

Carries out a similar procedure. After a job has finished, ESAMR calculates the job's stage weights on every node and saves these new weighs as a part of the historical information. Finally, ESAMR Applies k-means, a machine learning algorithm, to re-classify the historical information stored on every worker node into k clusters and saves the updated average stage weights for each of the k
Clusters. By utilizing more accurate stage weights to estimate the Time to End of running tasks, ESAMR can identify slow tasks more accurately than SAMR, LATE, algorithms. Reduces the runtime execution and achieves scalability but ignores data locality for launching backup tasks.

3.4. HAT (History-based auto-tuning Map Reduce Scheduling)**:** Quan Chen (2011) et al. [7, 13] suggested HAT scheduler for heterogeneous environments to execute straggler jobs. This scheduler calculates the progress of tasks more accurate than previous methods and adapts with different environments automatically. HAT uses historical information which is stored on each node to set the parameters and identifies slow tasks dynamically. Quan Chen et al. proposed several equations to calculate the weight of new phases, the progress score of map and reduce tasks, to detect slow tasks and slow nodes. Based on the accurate calculated progress of tasks, HAT estimates the remaining time of tasks accurately and further launches backup tasks for the tasks that have the longest remaining time. HAT estimates progress of a task accurately since it tunes the weight of each phase of a map task and a reduce task automatically according to the historical values of the weights. HAT, further classifies slow nodes into map slow nodes and reduce slow nodes. In this way, HAT can launch backup tasks for reduce straggler tasks on Map slow nodes and vice versa. It Increases system performance and achieves scalability [17]. But Ignores different weights for different job types and different dataset sizes and ignores data locality for launching backup tasks.

3.5. PURLIEUS:

B. Palanisamy (2011) et al. [8] developed a system called purlieus. It emphasized on locality in the shuffling phase and focused on pairing between the localization of tasks with VM and the data or resources. In this data is placed independent of the type of job processing it or the loads on the Servers, during data placement the following attributes are considered

i. Job Specific Locality-awareness: In this three distinct classes of jobs are used – (1) Map-input
Heavy (2) Map-and-Reduce-input heavy (3) Reduce-input-heavy

ii. Load Awareness: Placing data in a Map Reduce cloud should also account for computational load (CPU, memory) on the physical machines. A good technique should place data only on machines that are likely to have available capacity to execute that job; else remote-reads will be required to pull data from busy machines to be processed at less-utilized machines.

iii. Job-specific Data Replication Depending upon the type and frequency of jobs, it places each replica of the entire dataset based on a particular strategy. Improved data locality in this manner is beneficial in Two ways – (1) it reduces job execution times as network transfer times are big components of total execution time and (2) it reduces cumulative data center network traffic. But this system couldn't facilitate an end-to-end system optimization for data flow in MapReduce framework.

3.6. ADAPT (Availability –aware Map Reduce Data Placement for Non Dedicated Distributed Computing-2012):

Existing Map Reduce framework randomly distributes data blocks onto each node. This mechanism works well for typical cluster environment that assumes homogenous hosts. However, nodes in non-dedicated distributed computing environment are usually heterogeneous in both computing power and availability, which makes the existing random data placement strategy inappropriate. While the heterogeneity [19] in computing power has limited impact on data-intensive Map Reduce applications, hosts in non-dedicated computing environment may present considerable heterogeneous features in availability and hurt application performance. Hui Jin et al. [9, 13] proposed ADAPT, an *Availability-aware Data Placement* strategy for Map Reduce to mitigate the impact of volatility and heterogeneity. ADAPT dynamically dispatches data blocks onto participating hosts based on their availabilities. ADAPT helps to mitigate the impact of vulnerability without increasing the replication degree, improve the data locality of Map Reduce applications, and inherently reduce the network traffic. ADAPT significantly increases the data locality and reduces the data migration cost. Although the advantage of ADAPT in improving Map Reduce application performance is less significant for environment with higher network connectivity.

3.7. Maestro:

Shadi Ibrahim (2012) et al. [10] proposed a Scheduling algorithm for Map tasks to be selected based on the number of hosted Map-tasks and on the replication scheme for their input splits.

3.8. CoGRS:(Center-of-Gravity Reduce Task Scheduling):

In this the author has considered data locality, the partitioning kew problems and the network load is a special concern with Map Reduce as a large amount of traffic can be generated during the shuffle phase.it causes the performance degradation. *Mohammed hammoud et al* [11] proposed **CoGRS**.it is a locality-aware skew-aware reduce task scheduler

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 7, Issue 3, March 2020

for Map Reduce. CoGRS attempts to schedule every reduce task, *R*, at its *center-of-gravity* node determined by the network locations of *R's* feeding nodes and the skew in the sizes of *R's* partitions. The network is typically a bottleneck in Map Reduce-based systems [19]. By scheduling reducers at their center-of-gravity nodes, for reduced network traffic which can possibly allow more Map Reduce jobs to co-exist on the same system. CoGRS controllably avoids scheduling skew, a situation where some nodes receive more reduce tasks than others, and promotes pseudo-asynchronous map and reduce phases.

3.9. **Tarazu: optimizing Map Reduce on heterogeneous clusters".**

Faraz ahmad et al. [12] proposed **"Tarazu: optimizing Map Reduce on heterogeneous clusters".** In this the author has addressed the MapReduce's poor performance on heterogeneous clusters. The poor performance is due to two key factors: (1) the non-intuitive effect that Map Reduce built-in load balancing results in excessive and bursty network communication during the Map phase. (2) the intuitive effect that the heterogeneity amplifies load imbalance in the Reduce computation. Tarazu, a suite of optimizations to improve Map Reduce performance on heterogeneous clusters. Tarazu consists of (1) Communication-Aware Load Balancing of Map computation (CALB) across the nodes, (2) Communication-Aware Scheduling of Map computation (CAS) to avoid bursty network traffic and (3) Predictive Load Balancing of Reduce computation (PLB) across the nodes. Tarazu increases performance only for heterogeneous clusters not for homogeneous clusters. Jobby P Jacob[17] et al analyzed that using K-Means Clustering Algorithm when running on Hadoop Map Reduce on Eucalyptus [18] platform improves the network, memory bandwidth, data throughput and average I/O.

4. COMPARISON OF DIFFERENT SCHEDULERS:

An extensive literature survey has been made by going through the recent and relevant publications from the various Researchers in this field. The following are the major contributions already made by the researchers. Various features of schedulers are considered for comparison of Map Reduce Schedulers and also the Advantages and disadvantages various job scheduling methods are expressed in Table1.the following table demonstrates some enhancements can be made to improve the overall performance of Map Reduce.

Table 1. Comparison of Map Reduce Schedulers

Authors/ Year	***Method/ Approach***	**Parameters**							**Limitations**
		Runtime reduction	***Network reduction***	***Resource utilization increment***	**Suitable for heterogeneous environment**	***Scalability***	***Data locality***	***Satisfaction of user's high level performance goals***	
Matei Zaharia (2008)	*LATE*	✓15%		✓	✓				Ignores data locality for launching backup tasks
Quan chen et al(2010)	*SAMR*	✓24%		✓	✓	✓			Ignore different weights for different job types and different dataset sizes Ignore data

									locality for launching backup tasks
Quan chen et al(2011)	*HAT*	✓ 37%			✓	✓			Ignore different weights for different job types and different dataset sizes. Ignore data locality for launching backup tasks
Xiaoyn Sun et al(2012)	*ESAMR*	✓		✓	✓				Ignore data locality for launching backup tasks
Palanisamy et al(2011)	*Purlieus*	✓ 50%	✓ 70%				✓		Couldn't provide end-to-end utilization
Hui Jin et al (2012)	*ADAPT*		✓				✓	✓ 30%	Performance is less significant for high network connectivity
Palson et al	*MART*			✓	✓				-
Shadi Ibrahim(2012)	*Maestro*	✓34%	✓		✓				-
Hammoud et al	*COGRS*	✓3.2to6.3%	✓						Static sweet spot determination

Faraz ahmad et al	*Tarazu*				✓			✓	Network contention occurs among remote tasks on different nodes

5. CONCLUSION

Map Reduce has brought new excitement in the parallel data processing landscape. This is due to its salient features that include scalability, fault-tolerance, simplicity, and flexibility. Still, several of its shortcomings hint that Map Reduce is not perfect for every large-scale analytical task; it includes data locality, network reduction, scalability and response time. In this paper, some of approaches are presented along with their relative strengths and weaknesses. Also, some enhancements that can be developed by considering some of the parameters like data locality, scalability, network traffic, response time to improve the overall performance of the Map Reduce.

REFERENCES

[1] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. In Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation -Volume 6, pages 10–10, Berkeley, CA, USA, USENIX Association, 2004.
[2] Hadoop, http://lucene.apache.org/hadoop, June 30, 2012.
[3] Hadoop Distributed File System, http://hadoop.apache.org/hdfs
[4] M. Zaharia, A. Konwinski, A. Joseph, Y. zatz, and I. Stoica. "Improving map reduce performance in heterogeneous environments". In OSDI'08: 8th USENIX Symposium on Operating Systems Design and Implementation, October 2008.
[5] Q. Chen, D. Zhang, M. Guo, Q. Deng and S. Guo. SAMR: A Self-adaptive Map Reduce Scheduling Algorithm in Heterogeneous Environment. In Proceedings of IEEE 10th International Conference on Computer and Information Technology, 2010.
[6] X. Sun, "An Enhanced Self-adaptive Map Reduce Scheduling algorithm", Master Thesis, University of Nebraska, Lincoln, 2012.
[7] Q. Chen, M. Guo et al.,"HAT: history-based auto-tuning Map Reduce in heterogeneous environments", the journal of Supercomputing, pp.1-17, 2011.
[8] B. Palanisamy, A. Singh, L. Liu, and B. Jain, "Purlieus: locality-aware resource allocation for Map Reduce in a cloud," in Proceedings of ACM SC, pp. 58:1–58:11, 2011.
[9] H. Jin, Xi Yang et al , "ADAPT : Availability –aware Map Reduce Data Placement for Non-Dedicated Distributed Computing " , International conference in Distributed Computing Systems (ICDCS),2012.
[10] S. Ibrahim, H. Jin et al., "Maestro: Replica-Aware Map Scheduling for Map Reduce", Cluster, cloud and grid computing (CCGRID), 2012 12th IEEE/ACM International Symposium, pp.435-442, 2012.
[11] M. Hammoud , M. S .Rehman et al., "Center-of-Gravity Reduce Task Scheduling to Lower MapReduce Network Traffic",, Cloud Computing(CLOUD).2012 IEEE 5th International Conference,pp.49-58,2012.
[12] F. Ahmad, S. Chakradhar , A. Raghunathan , and T. N. Vijaykumar , " Tarazu : Optimizing MapReduce on heterogeneous clusters," in Proceedings of ASPLOS, pp. 61–74, 2012.
[13] Maedeh Mozakka, Faramarz Safi Esfahani, Mohammad H NadimiI, "Survey on Adaptive Job Schedulers in MapReduce" Journal of Theoretical and Applied Information Technology , Vol. 66 No.3,Aug 2014.
[14] A. Hemanth, Dr. R.V. Krishnaiah,"The Hadoop Distributed File System: Balancing Portability" International Journal of Computer Engineering & Applications,Vol. III, Issue III, July 2013 .
[15] Zhao Li · Yao Shen · Bin Yao · Minyi Guo," OFScheduler: A Dynamic Network Optimizer for MapReduce in Heterogeneous Cluster" springer October 2013.
[16] M. Berekmeri, D. Serrano et al "A Control Approach for Performance of Big Data Systems" 19 IFAC World Congress 2014.
[17] Jobby P Jacob, Anirban Basu et al "Performance Analysis of Hadoop Map Reduce on Eucalyptus Private Cloud" International Journal of Computer Applications, Volume 79 – No 17, October 2013.
[18] Tao Gu, Chuang Zuo,et al " Improving MapReduce Performance by Data Prefetching in Heterogeneous or Shared Environments" International Journal of Grid and Distributed Computing Vol.6, No.5, 2013, pp.71-82.
[19] Salma Khalil, Sameh A.Salem,et al "Mapreduce Performance in Heterogeneous Environments: A Review " International Journal of Scientific & Engineering Research, Volume 4, Issue 4, April -2013.
[20] Jongse Park, Daewoo Lee et al "Locality-Aware Dynamic VM Reconfiguration on MapReduce Clouds" 21st International ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC'12), June 18–22, 2012.
[21] Christos Doulkeridis · Kjetil Norvag,et al "A survey of large-scale analytical query processing in MapReduce " Published online:June 2013,Springer-Verlag Berlin Heidelberg 2013