INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
*WINGS TO YOUR THOUGHTS.....*

# Big Data Approach for Improving Execution Time in Heterogeneous Hadoop Environment

**Amandeep Kaur [1], Er. Sushil Lekhi [2]**

[1]Research Scholar, [2]Assistant Professor
Computer Science and Engineering, Punjab Technical University,
Rayat Institute of Engineering & Information Technology, Ropar, India
[1] amandeep.banwait@gmail.com, [2] lekhi.engg@gmail.com

***Abstract-*** *Big data has established an era of tera where large volume of data is being collected at fascinating speed. Due to hike in storage capabilities, processing power and availability of data, the size of global data is increasing in zeta-bytes. Hadoop is one of the famous technologies in the big data landscape for evaluating the data through Hadoop Distributed File System and Map-Reduce. Job scheduling is another important activity for proper management of cluster resources. In the proposed research work, SAMR algorithm has been improved in term of several aspects. The comparison has been performed on the basis of CPU Usage, Memory Usage, Total time consumption, Mapper time and Reducer time and it is evident from the simulation results that the Extended SAMR algorithm outperforms then the simple SAMR algorithm.*

***Keywords:*** *Hadoop, SAMR, ESAMR, Big Data, LATE and FIFO.*

## 1. INTRODUCTION

Big data has created an age of tera where enormous amount of information is being collected at interesting rates. Due to hike in memory capacities, processing influence and accessibility of information, the volume of global information is rising in zeta-bytes. Hadoop is one of the accepted technologies in the big data scene for assessing the information throughout Hadoop Distributed File System and Map-Reduce. Job arrangement is very significant activity for appropriate organization of cluster resources. Hadoop schedulers are pluggable elements which allocate assets to jobs. In different types of schedulers, in style are the FIFO, Fair as well as Capacity schedulers.[1]

MapReduce is an encoding paradigm and an linked operation for exchanging and producing large datasets. It permit users to state a map utility which courses a key/value couple to make a set of midway key/value couple, and a diminish utility that unites all the midway standards linked with the similar midway key. Map Reduce was initiated by Google, in union with GFS and Big Table comprising spine of Google's Cloud Computing platform [2]. Map Reduce has attained immense success in diverse applications including from straight and vertical exploration engines to GPU to multiprocessors. MapReduce has been believed as one of the key empowering methodologies for striking care of ceaselessly rising requests on outlining benefits required by Big Datasets yet at the similar time lots of questions arrive with MapReduce keeping in brain the ending goal to hold a much supplementary extensive cluster of employments, mixture into Hadoop's native file system. The intention following this is the lofty versatility of the MapReduce worldview which believes hugely parallel as well as circulated implementation over a costly number of figuring hubs. [3,4]

## 2. THREE V'S IN BIG DATA

**1. Volume of information**: Volume refers to ability or the amount of information. Volume of information stocked in enterprise repositories has amplified from megabytes, gigabytes to pet bytes.

**2. Variety of information**: It refers to dissimilar forms of information and resources of information. Information diversity increased since prepared and legacy information stored in venture repositories to shapeless, semi prepared, audio, video, XML et cetera.[5]

**3. Velocity of information:** Velocity refers to the swiftness of information processing. For time-sensitive procedure for instance catching racket, big information should be implemented as it brooks into our venture in order to exploit its worth.

## 3. EXISTING PROBLEMS WITH BIG DATA PROCESSING

- **Assorted and Incompleteness:** When person utilize information, a huge agreement of heterogeneity is contentedly suitable. In detail, wealth of natural speech can supply precious deepness. Though, device analysis algorithms deem homogeneous information, and cannot appreciate nuance. As a consequence, information should be cautiously prepared as a primary footstep in (or proceeding to) information examination. Computer schemes work mainly efficiently if they can stock up manifold items that are all similar in dimension and arrangement. Proficient representation, access, and study of semi-structured information need additional work in a competent manner.[6,7]
- **Range:** The incredibly primary thing anybody thinks concerning Big Data is its dimension. After all, the phrase "big" is there in the very name. Managing enormous as well as rapidly increasing amount of information has been a demanding job for lots of decades. In the history, this confront was attained during processors getting quicker, subsequent Moore's law, to supply us with the possessions required to manage with increasing quantity of information. But, there is a

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY

*WINGS TO YOUR THOUGHTS…..*

basic shift in progress now: information volume is scaling quicker than calculate resources, and CPU speeds are steady.

- **Appropriateness:** The other part of dimension is speed. The larger the information set to be considered, the longer it will obtain to analyze. The design of a scheme that economically deals with dimension may too consequence in a scheme that can operate a given dimension of information set sooner. Though, it is not only this hustle that is typically meant when one speaks of swiftness in the situation of Big Data. Rather, there is a information gathering rate confront.
- **Confidentiality:** The confidentiality of information is another significant concern, and one that goes up in the situation of Big Data. For electronic strength records, there are firm laws leading what can and cannot be made. For other information, policy, mainly in the US, rule is flexible. Though, there is community panic regarding the erroneous exercise of individual information, mostly during linking of information from many sources. Managing confidentiality is effectively together a technological and a sociological trouble, which should be considered jointly from both perspectives to understand the pledge of big data.
- **Human being teamwork:** Despite of the number of progress made in computational examination, there is lots of pattern that person can effortlessly recognize but workstation algorithms have a firm time judgment. Mostly, analytics for Big Data will not be all computational somewhat it will be intended openly to have a person in the loop. The newest sub-field of visual analytics is trying to do this, at least with admiration to the modeling as well as examination part in the channel. In today's self-motivated world, it frequently takes several experts from diverse areas to actually locate out what is going on. A Big data examination arrangement should sustain contribution from numerous human experts, and shared examination of consequences. These specialists might be departed in space and instance when it is too expensive to gather a complete squad jointly in one room. The data scheme has to take this distributed specialist input, and sustain their teamwork.

## 4. SCHEDULING ALGORITHMS

**FIFO scheduler** is the popular scheduler of Hadoop support on the idea that the jobs are implemented in the command of their surrender. The Job Tracker selects the oldest job first from the job queue. This scheduler cause hunger to little jobs whereas longer jobs end in reasonable instance precedence to the jobs which required being finished in a timely way lacked in this scheduler. This scheduler is well-organized and easy to apply and the charge of whole scheduling procedure is also fewer But it is intended only for solitary kind of job and resulted into little presentation when organization manifold types of jobs.

**Fair scheduler** was extended at Facebook in a method to allocate resources to the jobs such that every job in the group gets fair portion of resources in excess of time. Thus here jobs are owed an equal quantity of resources as a consequence of which diversity of jobs which obtain both extended and little time to complete end in an intermixed way maintaining a equilibrium amongst the resources of the group. It was less composite and works healthy with mutually small and big cluster. It does not judge the job load of every node.[8,9]

**Capability scheduler** was intended for large clusters in the association. Unlike Fair Scheduling where groups are formed, this scheduler generates queues of certain ability with the configurable figure of plot and decreases slots. This scheduler in general utilizes the group ability among the client and can be configured within the Hadoop arrangement files provided by the Apache Hadoop packages.

## 5. PROBLEM STATEMENT

In previous work Self-Adaptive MapReduce scheduling algorithm (SAMR) has been implemented. SAMR: a Self-Adaptive MapReduce scheduling algorithm, which estimates improvement of tasks vigorously and acclimatize to the constantly altering surroundings automatically. SAMR is implemented with a similar thought to LATE MapReduce scheduling algorithm. Though, SAMR obtain enhanced PS standards of all the tasks by means of past data. By utilizing correct PSs, SAMR disclose actual slow tasks in addition to decreases implementation time evaluate with Hadoop and LATE.[11]

### 5.1. SAMR ALGORITHM

1: method SAMR
2: input: Key/Value couples
3: output: Statistical end result
4: Reading past data and tuning parameters with it.
5: Identifying slow tasks
6: Identifying slow Task Trackers
7: beginning backup tasks
8: Gathering results and updating past data
9: end method

### 5.2. DRAWBACKS OF SAMR ALGORITHM

- It finishes the tasks slow.
- It does not consider that the dataset dimension and the job kind can also influence the stage weights of map and decrease tasks.

## 6. PROPOSED ALGORITHM

In the proposed research work, Extended SAMR algorithm has been implemented in heterogeneous Hadoop Environment. Extended SAMR algorithm is intended to conquer the limitation of SAMR algorithm by considering several factors that might blow the stage weights. The major action engaged by ESAMR is to categorize the past data stored on every TaskTracker node into k clusters by means of a machine learning method. ESAMR concern K-means, a machine learning method, to re-classify the past data stored on every TaskTracker node into k groups and accumulates the standard stage weights of every of the k groups. By utilizing additional correct stage weights to calculate approximately TimeToEnd of every task, ESAMR can identify slow tasks additional precisely than SAMR as well as LATE algorithms.

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

**Table: 1** Types of scheduler [10]

| S. No | Type of Scheduler | Algorithm |
|---|---|---|
| A. | Data Locality Aware Schedulers. | 1. LARTS<br>2. CoGRS<br>3. CASH<br>4. Improving data locality of MapReduce by Scheduling in uniform Computing Environments.<br>5. Distribution Aware Scheduling ways<br>6. TDWS |
| B | Speculative Execution based Schedulers. | 1. LATE<br>2. SAMR<br>3. HAT<br>4. ESAMR |
| C | Resource Contention aware Schedulers | 1. Toward a Resource aware scheduler in Hadoop<br>2. Job Aware Scheduling Algorithm for MapReduce.<br>3. Load-Driven Task Scheduler with Adaptive Dynamic Slot Controller (DSC).<br>4. LsPS<br>5. A new scheduler strategy fo heterogeneous workload-aware in Hadoop.[13] |
| D | Performance Management based Schedulers | 1. Performance Driven Task Co-Scheduling for MapReduce structure.<br>2. Power Management of accelerated MapReduce Workloads in Heterogeneous groups.[14] |
| E | Energy and Makespan aware Schedulers | 1. Energy proficient thermal aware task scheduling for homogeneous high computing data centres.<br>2. Thermal and Power-Aware Task for Hadoop based storage centric data storage.<br>3. Temperature, Power, and Makespan Aware Dependent Task Scheduling for Data Centres.<br>4. Power-aware Scheduling of Mapreduce applications in the cloud.[15] |

**Algorithm: ESAMR**

Require:
1: PFM (Percentage of Finished Map Tasks), a threshold considered to manage when to start the slowmap task recognition
2: PFR (Percentage of Finished Reduce Tasks), a threshold considered to manage when to start the slow diminish task recognition
3: history, past information of the K clusters, where each confirmation of a cluster contains 5 standards, M1, M2, R1, R2 and R3
4: threshold, a variable for choosing slow tasks
5: Main Process
6: if a job has finished PFM of its map tasks afterwards
7: M1= ComputeWeightsMapTasks
8: M2=1-M1
9: end if
10: if a job has finished PFR of it's reduce tasks afterwards
11: < R1;R2 >= ComputeWeightsReduceTasks
12: R3=1-R1-R2
13: end if
14: slowTasks= locateSlowTask
15: run backup tasks used for slowTasks
16: if a job has ended then
17: run K - means method to re-classify past information into k clusters
18: end if
19: Method ComputeWeightsMapTasks
20: if a node has ended map tasks for the job afterwards
21: compute tempM1 based on the job's diagram tasks finished on the node
22: M1=arbitrarily chosen first phase weight M1 from the equivalent node's history
23: beta=abs(tempM1-M1)
24: for each M1[i] 2 the node0s record, i=1.2,...,K do
25: if abs(M1[i]-tempM1)<beta then
26: M1=M1[i]
27: beta=abs(tempM1-M1[i])
28: end if
29: end for
30: return M1

## 7. RESULTS

The parameters considered for comparison are CPU usage, Memory usage, Total Time Consumption and Execution time.

**CPU usage:** - CPU utilization refers to a computer's utilization of processing resources, or the quantity of job hold by a CPU. Genuine CPU use varies depending on the quantity and kind of administered divided tasks. Assured tasks need heavy CPU instance, whilst others need less because of non-CPU resource needs.

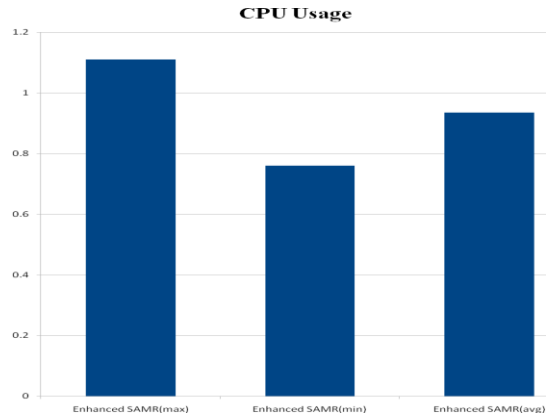


**Figure 1**: CPU utilization by ESAMR algorithm

**Memory Utilization:** Memory consumption is considered as the addition of memory utilized by all procedures. This is next divided by the entire Memory of the Server as well as multiplied by 100 in command to show as proportion. So, Memory use = (summation of Memory utilized by all processes / entire Memory) * 100

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

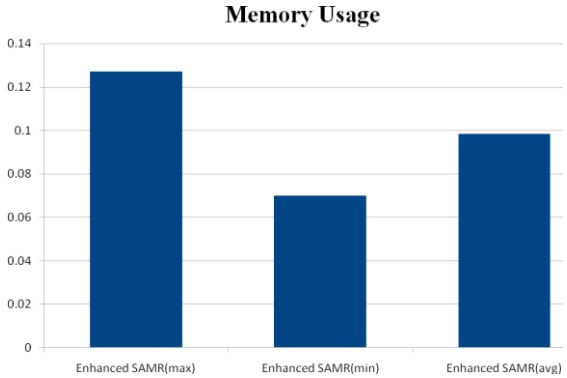


**Figure 2**: Memory utilization by ESAMR algorithm

**Total time consumption**: Total CPU time is the summation of CPU time consumed by all of the CPUs utilized by the computer program.

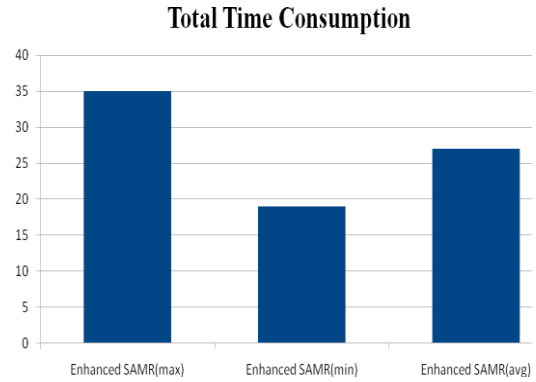


**Figure 3:** Total time consumption by ESAMR algorithm

**Execution time**: The execution time is defined as the time spent by the system executing that task, including the time spent executing run-time or system services on its behalf.

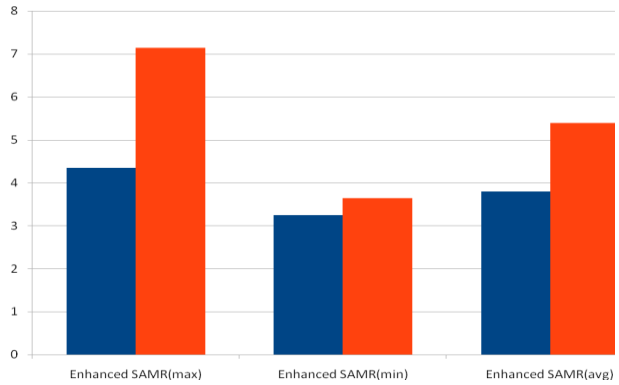


**Figure 4**: Execution time of ESAMR algorithm

## 8. FUTURE SCOPE

In future, the performance of various scheduling algorithms can be improved further. There are number of parameters that are still not considered for comparison between different algorithms. There are still many phases that need to be considered such as Data locality, backup task scheduling and load distribution.

## REFERENCES

[1] Q. Chen, "SAMR : A Self-adaptive MapReduce Scheduling Algorithm In Heterogeneous Environment," no. Cit, 2010.
[2] M. Brahmwar, M. Kumar, and G. Sikka, "Tolhit – A Scheduling Algorithm for Hadoop Cluster," Procedia - Procedia Comput. Sci., vol. 89, pp. 203–208, 2016.
[3] X. Sun, C. He, and Y. Lu, "ESAMR : An Enhanced Self-Adaptive MapReduce Scheduling Algorithm," 2012.
[4] A. Rasooli and D. G. Down, "A Hybrid Scheduling Approach for Scalable Heterogeneous Hadoop Systems," pp. 1284–1291, 2013.
[5] L. Li, Z. Tang, R. Li, and L. Yang, "New improvement of the Hadoop relevant data locality scheduling algorithm based on LATE," pp. 1419–1422, 2011.
[6] S. Su and P. Gopalan, "An Optimal Task Selection Scheme for Hadoop Scheduling," IERI Procedia, vol. 10, pp. 70–75, 2014.
[7] P. C. Science, A. Spivak, and D. Nasonov, "Data Preloading and Data Placement for MapReduce Performance Improving," Procedia - Procedia Comput. Sci., vol. 101, pp. 379–387, 2016.
[8] M. Vaidya and S. Deshpande, "Critical Study of Performance Parameters on Distributed File Systems using MapReduce," Procedia - Procedia Comput. Sci., vol. 78, no. December 2015, pp. 224–232, 2016.
[9] S. Maitrey and C. K. Jha, "MapReduce : Simplified Data Analysis of Big Data," Procedia - Procedia Comput. Sci., vol. 57, pp. 563–571, 2015.
[10] B. Memishi, "Diarchy : An Optimized Management Approach for MapReduce Masters," vol. 51, pp. 9–18, 2015.
[11] G. W. Cassales, A. S. Char, M. Kirsch, and L. A. Ste, "Context-Aware Scheduling for Apache Hadoop over Pervasive Environments," vol. 52, no. Ant, pp. 202–209, 2015.
[12] N. Srinivas, A. Negi, and V. N. Sastry, "Performance Improvement of MapReduce Framework in Heterogeneous Context using Reinforcement Learning," Procedia - Procedia Comput. Sci., vol. 50, pp. 169–175, 2015.
[13] J. Xie, F. Meng, H. Wang, and H. Pan, "Research on Scheduling Scheme for Hadoop clusters," Procedia Comput. Sci., vol. 18, pp. 2468–2471, 2013.
[14] V. Subramaniyaswamy, V. Vijayakumar, R. Logesh, and V. Indragandhi, "Unstructured Data Analysis on Big Data using Map Reduce," Procedia - Procedia Comput. Sci., vol. 50, pp. 456–465, 2015.
[15] T. D. Plantenga, Y. R. Choe, and A. Yoshimura, "Using Performance Measurements to Improve MapReduce Algorithms," Procedia - Procedia Comput. Sci., vol. 9, pp. 1920–1929, 2012.