ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

Analyzing & Optimizing Spark Performance

Shaik Hussain Bhasha[1], Dr. G.A.Ramachandra[2]

Research Scholar, Dept. of CST, S.K.U, Ananthapuramu, India[1]

Professor, Dept. of CST, S.K.U, Ananthapuramu, India[2]

ABSTRACT: Actually Analyzing and processing the SPARK in timely and cost effective manner is very prominent approchement and its very hectic job. And as usually the SPARK makes the huge data to understand in easy way that can be happened (visualized) in a snap deal. In the market so many tools available for making explored such things, but one of the best tool is 'SPARK' which makes the data to be adopted for repository and managing the things of 'BIGDATA'. So many methods were suggested and developed for analyzing and augmenting 'SPARK' accomplishment (performance).In my paper piercely being focused on tuning configuration parameter approachement. Even though several parameters those affect the performance of 'SPARK', in which map reduced related parameters made on effective impact. The main aim of this work to raise the overall performance of 'SPARK' by reducing the job execution time. Reduction in time is attained through tuning some of map reduce associated parameters. To get understood these parameters is paramount as different types of parameter with clumsy(improper) values which shows negative input upon overall performance. In this paper what we proposed method is that saves the job execution time and optimizes disk usage properly and effectively. It exactly improves the overall performance of the 'SPARK' by 38.53% over the base system in heterogeneous environment.

KEYWORDS: Spark; Hadoop; BigData ; MapReduce; Configuration Parameter; Performance Optimization.

I. INTRODUCTION

Advanced data innovation and computerized world results in producing a huge volume of data known by 'BigData'. These data can classified into 3 types which are

1) Structured data
2) Unstructured data
3) Semistructured data.

In earlier the data management techniques commonly focused upon structured data and unable to processing the unstructured data. But now a days a huge and gigantic part of 'BigData' is utterly unstructured data. Which can be proceed to retrieve the meaningful raw data. And basically the traditional and conversational were fairly inept to deal with the factors such as large volume, large velocity and a different kind of big data. To find the problem of traditional and conversation management techniques, several tools were developed in which one of the best and open source tool is preprocessed by apache is known as 'SPARK'. Apache SPARK[1] is a open source tool. It is Java/Python/Scala based programming platform, which is designed for working with the large amount of structured and unstructured data in a distributed computing world. Actually SPARK – open source tool which fascinates many researchers to make use of different techniques and improve the quality to deal with the 'BigData' more accurately and acutely. It will process the data and extract the meaningful data which can be explored/used by the averages to enhance their productivity. 'SPARK' components are 1) HDFS 2) MapReduce 3)Spark Core 4)Spark SQL 5)Spark Streaming 6) MLib(Machine Library)7)GraphX(Graph) and HDFS store the data in the form of Blocks, while MapReduce can be used for access the data stored data. SPARK delivers more throughputs when it comes to Big Data application. It can handle all the type of tasks such as large datasets, performance computation and present the o/p to the client in real time environment. SPARK frame work consume resources like CPU utilization, memory buffer etc., in order to give a best and high performance. But in real time scenario it's not possible to make use all of them at a time in a optimize manner due to

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

many factors like multiple job execution, lack of knowledge parameters etc., these are many possibilities are developed for optimizing SPARK- performance. One method is based upon configuring the 'SPARK parameter'. SPARK has several configured files those consisting of various parameters which can have great significance on SPARK execution. This paper proposes tuning of the configuration parameters for a particular MapReduce job in a cluster. SPARK configuration depends on several factors; those are cluster size, the configuration of nodes, data size, execution environment, available resources, memory buffer, disk space, etc. A variation in the parameter may be different according different applications and its environment.
For an instance, the block size may vary 64 MB to 128MB or assigned heap memory may increase for a huge amount of data, or sometimes the lower values of same parameters can be affording good results.
For the performance enhancement, it is very difficult to build an acceptable model that can be applicable all types of jobs and cluster configuration. In my work an optimization can be defined for the task by selecting and tuning some of the important parameter of Map Reduce model. At my work this experiment diminishes the execution time, increase CPU utilization and enhances the 'disks proper usage' by increasing number of reducers slots and heap memory size. The rest of the paper propagates as follows – **Section-II** discusses the related work of this field, **Section III** explains the internal architecture of SPARK, **Section IV** focus on the methodology, **Section V** includes implementation and evaluation, and finally **Section VI** concludes the paper and future scope.

II.RELATED WORK

On this part which means SPARK accomplishment/performance on which many researchers and several students had took place at different levels. Some of research being gone through on developing 'job scheduling algorithm' for map reducing frame work, other optimizing performances by meta data availability and storing additional metadata table of relating jobs. Several groups being tunned the configuring the parameters to improve the performance and other make explored the things of execution mechanism of Map Reduce framework and target those sections which decrease / degrading the performance, meant by 'shuffle and sort phase'; other categories developed their frameworks by changing some features of traditional SPARK.

In 2011 Mohammad Hammoud et al.[2] proposed LARTS**(Locality-Aware Reduce Task scheduler)** which is a real time scenario that can schedule for improving map redcuing performance. In order to that, ZhuoTang et al.[3] who develops a scheduling methods/policy that lower the start time of reduce phase of MapReduce in various job environment.

In 2013, Rong GU et al.[4] develops SHadoop to reduce the execution time , by introducing instant messaging communication technique and optimize setup and cleanup task at beginning and ending of the job.
Self tuning models were also developed for performance environment for Big Data, 'StarFish falls into this category. Spark architecture that provides good performance as per new requirements and workload of the *sustain* ; introduced by Herodots Herodotu. SPARK is prepared by HAmoud also improves SPARK performance by storing the additional metadata *table of* relating jobs of DNA sequence.

When you start to computing the huge data, it will take more time for the shuffling and sort phasing of map reduce; So this type of taking more time of action reduced and proposed by yanfeci Guo et al.[7] in ishuffle. And actually by varying the configuration parameter, it can enhance performance of SPARK, which proposed by kewen wang & swathi prabhu . Initially the former group introduced 'the predator' which is a manual (experience guide) for optimizing configuration parameter and classified the parameters into groups. The latier group changes the tunable parameters of MapReduce, which refers that increases the number of mapper and reducer slots, buffer memory and compress the mappers output, those helps to get the high performance/accomplishment at homogeneous environment. But the leading organization like Intel & AMD were also presented their white paper at out SPARK configuration tuning. When we comparing and contrasting with earlier studies and proposals, in our paper we better improve stable performance with the assistance of configuration parameters in the heterogeneous environments. Actually the selecting

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

of the parameters have done, in this way that varies these with certain values those helps, increase its performance not withstanding any data sets, jobs and its working environments.

III.INTERNAL ARCHITECTURE OF SPARK

A. SPARK

SPARK is an open source computing frame work that specialized at 'data analytics'. It builds on the top of 'Hadoop HDFS'. The parallel abstraction of MapReduce inspects the programming model of the 'SPARK' Fig:1 below represent the framework of Apache SPARK. It keeps favorable properties of MapReduce, those are parallelization, fault tolerance, data distribution and load balancy, if (SPARK) affords a super for iterative classes of algorithms interactive applications and algorithms contain common parallel primitive methods, such as JOIN and Match.
SPARK leads the prominent/vital data into the cluster memory, based on users application and if applies the computations directly into memory for making faster the process than Hadoop process. Its quite opposite to Hadoop, which means it requires, loading the necessary data from HDFS in every iteration. And mainly SPARK Keeps the data in the memory, in between iterations; by this mechanism 'SPARK' is an apt one for algorithms those iterates the data. Fig:2 below represent the model of Apache SPARK shows the reason, how it is being gained the popularity on its own.

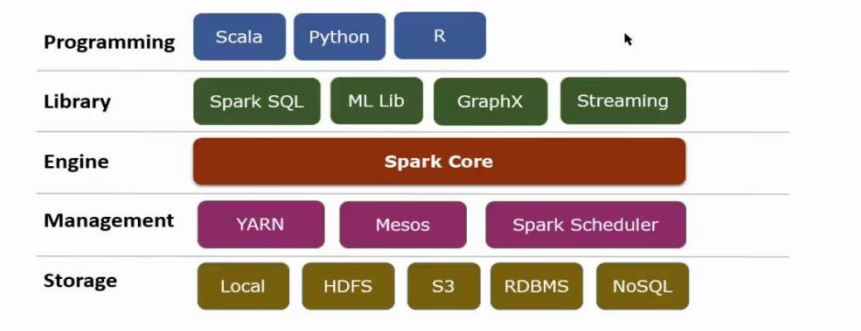

Fig.1. Spark Framework

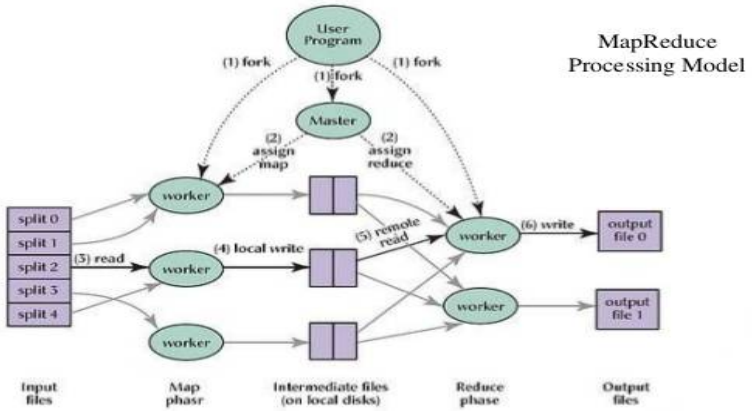

Fig.2. Apache Spark Model

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

B.HDFS

Actually HDFS is the reliable high bandwidth, low cost data storage and highly scalable cluster that facilitates the management related files throughout across machines. It casually follows Master-Slave concept and architecture. By default HDFS provides 64 MB to 128MB of Block Size to store data. HDFS framework that consists of Name Node and Data Node, Fig:-3 represent the HDFS architecture clearly.

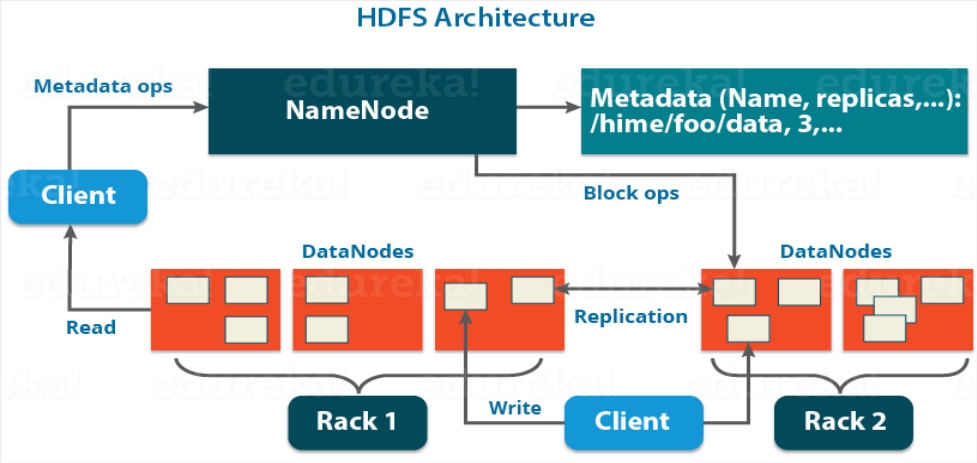

Fig.3.HDFS Architecture

Name Node:

It can call as Master Node as well it performs some of the operations like, opening & renaming and closing the files and its directories. It stores metadata of the files. The Meta Data comprises of very important data like block location, block name etc., but only one Name node is present therefore it is unique at SPARK environment.

Slave Node:

The data node is also called "Slave Node". Actually it performs some of the like block creation,deletion and replication as well. Whenever the instructions received from the 'NameNode' it ensure that it establishes /creates 'Heat Beat' messages to delete and connectivity between both nodes. And prominently some of the factors are there , which specifically associated with HDFS and determine /derives block size;DataNode IPAddress, replication etc throughout the cluster.

C. Map Reduce Engine:

MapReduce engine will have a ability to break the data into a meaningful manageable chunks and process the data on the distributed cluster commonly.

Actually the 'MapReduce' name originated from the effective of two capabilities, like Map and Reduce.

'Map' which works/tasks on data blocks in parallel and each task return a stipulated output as well.

'Reduce' tasks which receive these map outputs as it inputs and process them to produce the final outcomes (results)

Map and Reduce both engines do their work promptly and very transparent to the client towards its progress.

Fig: 4 expalin briefs the MapReduce architecture.

(1)Map Phase:- Each Mapper would be assigned an input split and it would generate the key value-pair. The number of mappers can be determined by the number of input splits and casually, mappers are equal to their input splits.

Number of Mapper= Number of input splits

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

After a Key-value pair generation output will be buffered at 'circular memory'. If the memory reaches its threshold point; then spill files will be generated and merged. After merging, at copy phase the output of mapper will be copied and make available to the reducers.

(2) Reduce Phase:- At first, the mapper outputs will be sorted and produced the intermediate results. The intermediate outputs merged together and makes a final output present to the client.

And there are some of the factors/parameters are associated with the MapReduce, which shows a direct impact on the Map and Reduce phase. Eg: Map Tasking, Number of Reduce tasks.

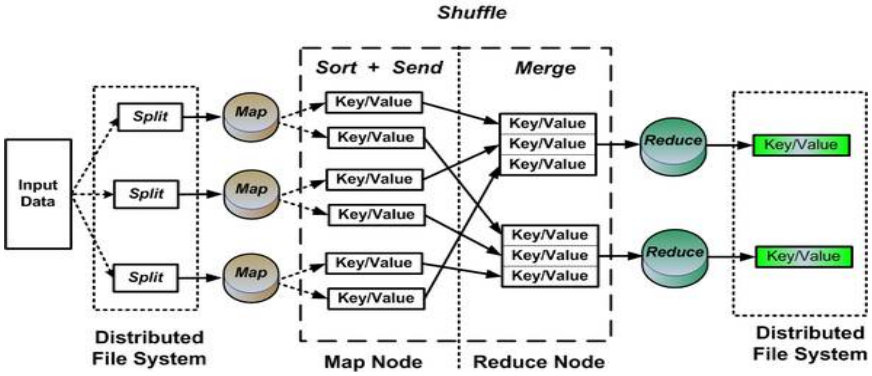

Fig.4. MapReduce Architecture

D. YARN:

This is the factor which works on top of the MapReduce framework. It allocates/distributes the resources among the data nodes for job scheduling and it manages multiple jobs in the cluster. YARN looks above fig No:5

It will have two components which are

1) Resource manager
2) Node Manager

1) Resource Manager:- it will take care of each and every job assigns to Data Node.

Whenever the MapReduce job is sorted, it creates/generates an 'application master' to take care of the job life cycle.

It dwells at on 'Masterside' or sometimes 'Name Node' itself behaves / configures as Resource Manager. Scheduling policy adapts to far and improper allocation of the memory buffer, CPU, disk capacity, IO buffer etc.,

2) Node Manager:

The Node manager supervises every task in the cluster. These are so many Node manager works for the proper functioning of tasks. Actually Node manager phase on slave.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

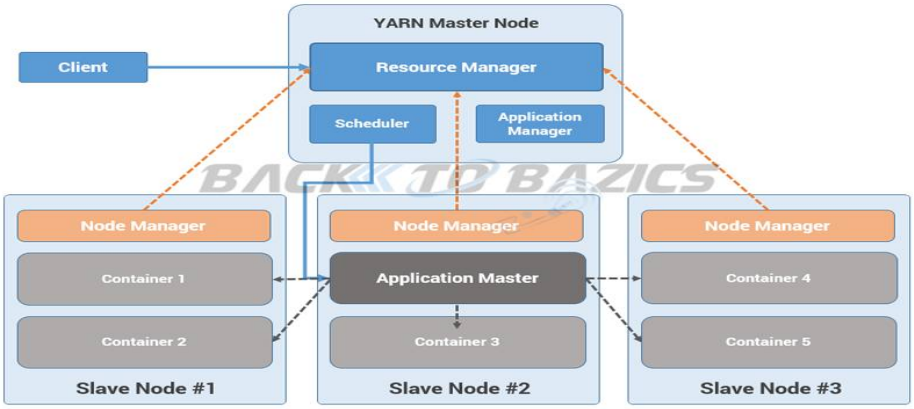

Fig.5. YARN Components

IV. METHODOLOGY

There are several categories of parameters/factors are present at SPARK. Actually these factors blended with core, YARN, HDFS. Atmost we must take care when we selecting the parameters, because each parameter would have assignment impact on the job execution. Varying these parameters/factors value may result on a positive or negative impact on the job execution time. This section briefs the process of parameters that tunes for a specific datasets and jobs. The fig.6 portrays the flowchart of tuning process. In this scenario the first performance measurement takes place with the default value; so it is called as "Base System". Then, alter/give another altered parameter value and again observe the performance by running the for the same data sets and check the altered value is **apt or not.**
If it is yes it becomes the final value for that configuration if not repeat the process again. This experiment utterly focuses on '3' factors/parameters related to 'MapReduce Engine'. All other parameter would have its efficiency irrespective of any kind of job and data sets. These parameter gives the approximately to process any job, and cluster configuration towards performance enhancement.

These are the steps for making optimized techniques are

1) Run the wordcount job with default configuration setting to get the overall system performance. It is known as base system.
2) Change and tune the concern configuration parameter and run the job again.
3) Compare the results with the base system and run the job until we get the optimum results.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

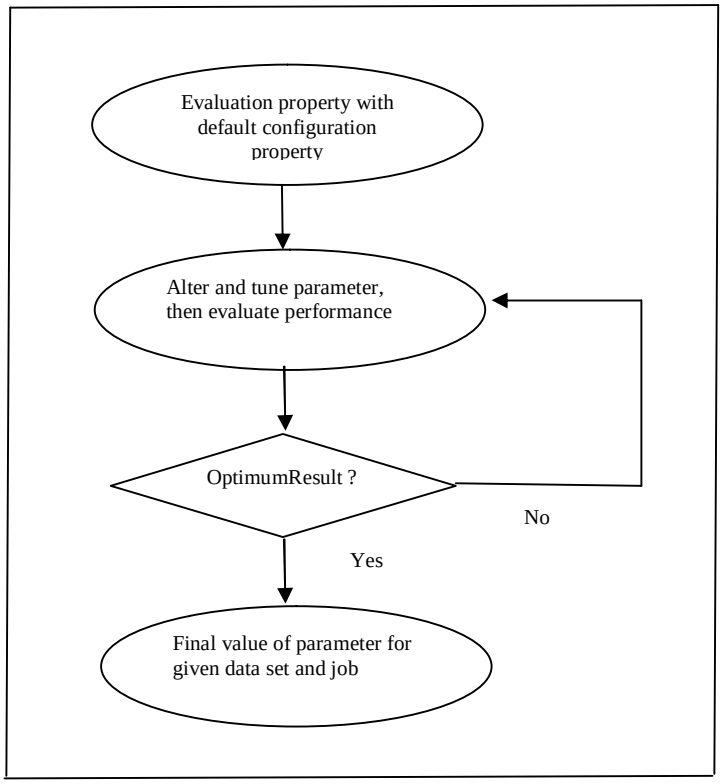

Fig.6. Flow chart of tuning technique

A) Impact of increasing the number of reducers container:

At the time of job execution, the number of Map & Reduce containers are assigned for parallel processing to gain high performance. Assignment of Mapper and Reducer containers Utterly/fairly depends on CPU cores and memory heap size available in the concern respective cluster environment. It is very important one in the perspective of heterogeneous cluster nodes to utilize maximum available resources to get a high performance of MapReduce job. Its very crucial point to understand the resources for which gets optimum results. “mapreduce.tasktracker.tasks.maximum” factors/parameters defines the value of reduces container will be tuned on “mapredsize.xmlConfiguration” file. It refers the maximum number of reducing tasks can be run simultaneously on Node; which highly recommended in the cluster that for optimum outcome/result, and every node would have same task to execute. Hence this value is set, in this way that every node should process some proper functioning, like Map or Reduce. The number of mapper may also increase, if mapper takes more time than reducers. So suggesting value for this configuration can be defined by an equation given below.

$$(Cores/2) < \text{number of reducers slot} < (cores*2)$$

(B)Impact of increasing sort factor for merging:-

Each mapper will produce the intermediate output that can be stored at circular memory buffer. Whenever this buffer reaches the concern predefined threshold, contents start spilling to disk by a background thread. Before writing the spills to the disk, the data will be divided into partitions/segments by background thread as per the intended resources where they are sent to be. And within every partition process background thread performs the sorting operation and if

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

the container feature enables then it runs on the sorted output. "mapreduce.task.io.sort.factor" is tuned at **mared-site.xml** and it briefs the number of sorting factors. This parameter derives " when sorting files number of streams to be merged at once" And greater value allows more streams to compile in a single pass and significantly improves the performance."mapreduce.task.io.sort.mb**"** benefits the memory buffer value in MB's for sorting data and it limits the sort factor value as well.

(C)Impact of increasing I/P for merging:-

As early as possible, mapper functions its task, the outputs of mapper transferred or copied to JVM memory of reducing task. To sorting this data for shuffling, some of the buffer space will be resolved at heap memory.
"mapreduce.reduce.shuffle.input.buffer.percent" that is the parameter/factor which briefs the input buffer shuffling. The above factor/parameter briefs " how much memory is being allocated to the storing map output during the shuffle" Heap memory bounds the value of input buffer for shuffling.

V. IMPLEMENTATION & EVALUATION

A) Environmental setup and performance evaluation:

There are 3 kind of are there as a cluster, which is a heterogeneous based.
Cluster will be implemented on Windows with Apache SPARK 2.1.0 stable release version. JDK 1.8 and Ssh will be need to manage SPARK daemons. Wordcount job execution will be worked on Twitter asset departure data set of size 6.5GB. Initially executes the word count job with default configuration property, after that again run the word count job with the optimizing values of each parameter and compare outcomes of each proposed system individually with base system. At eventually all the parameter values can be changed together and compares it values with the base system. SPARK with default configuration value completes the wordcount job over 6.5GB data in the fraction of seconds(2206). Table 2 affords the last parameter with theirs default and optimized values.

B) Impact by increasing number of Reducers container.

"mapreduce.tasktracker.reduce.task.maximum" parameter specify adjust the number of "reducer slots"
Equation (1) briefs the values will be **specifies**

$$\{(2/2) =/\sim (2*2)\} = \{1,4\}$$

The default value for this is "2" and altering value is "3". So that total slots are 6. Because 2 slaves; hence it called as proposed system1. It will enhance/increase the SPARK performance/accomplishment by 27.65% on the base system. Since it took a few seconds (1596 sec) comparison between base system and proposed system 1 portrays on fig :- 7

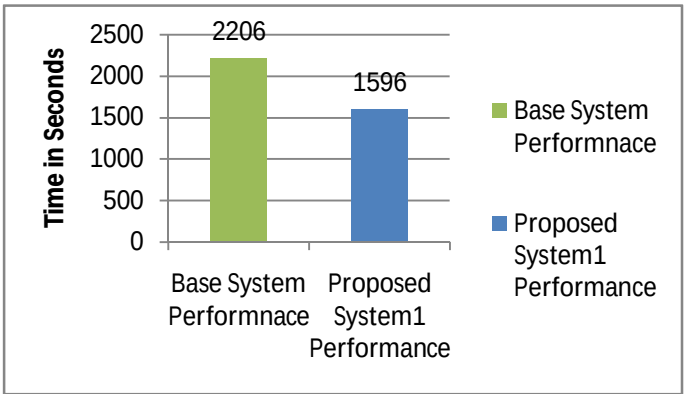

Fig.7.Comparison Chart between Base System and Proposed System1

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

C) Impact by increasing sort factor:

Enhance/augment the number of sorting streams to merge at once at the time of sorting. Its value being defined by "mapreduce.tsk.sort.factor". Its default value of sorting factor would be 100 and proposed value would 400; so more streams merged at once, and make reduced the execution table.

TABLE I: HARDWARE PROPERTY OF EXPERIMENT ENVORONMENT

Node	Hardware property
Master (NameNode)	Inter® core™ i3-2328M CPU @2.20GHZ,4GB RAM,32bit dual core processor
Slave1 (DataNode1)	Inter® core™ i3-330M CPU @2.13GHZ,4GB RAM,64bit dual core processor
Slave2 (DataNode2)	Inter® core™ i7-3770 CPU @3.40GHZ,4GB RAM,64bit dual core processor

TABLE II: PROPSED AND IMPROVED PARAMETER LIST

Parameter	Default Value	Optimized Value	Name
mapreduce.tasktracker.reduce.tasks.maximum	2	3	Proposed System 1
mapreduce.task.io.sort.factor	100	400	Proposed System 2
mapreduce.reduce.shuffle.input.buffer.percent	0.7	0.9	Proposed System 3

Sorting factor will be limited by the memory buffer allocates for sorting the data.

Apache SPARK – 2.1.0 mapreduce.task.io.sort.mb" value will be 512MB. So the proposed system calls by proposed system 2. And it enhance the SPARK performance by around 46.32% over base system it takes only(1184) a few seconds shown in fig:-8

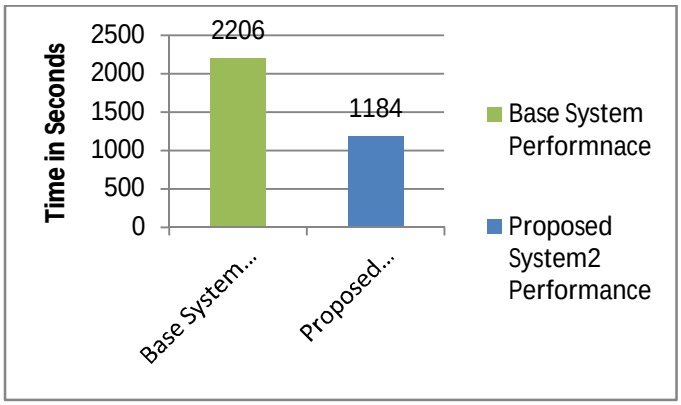

Fig.8.Comparison Chart between Base System and Proposed System2

D) Impact increasing I/P buffer for shuffling

During the copy phase of shuffling, enhance the percentage of reducers task heap would be needed for buffering outputs. Suppose want to increase the execution's speed of job. Just set the value to 0.09 that refers it will use the 90% of heap buffer; the default value of this parameter is 0.7 which specified by the

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

"mapreduce.reduce.shuffle.input,buffer,percent" .It improves the SPARK performance with 30.82% over base system . since it took only a few seconds (1526 sec) figure in fig 9. So it can be known as proposed system 3.

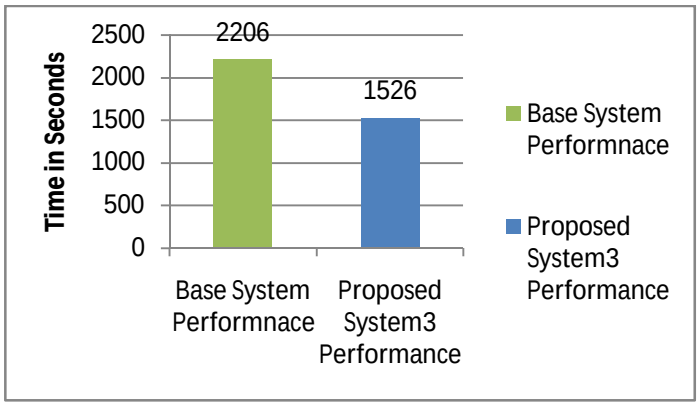

Fig.9.Comparison Chart between Base System and Proposed System3

E) Overall performance of improved system:-

Change the all parameter values with the optimized values and checks the overall performance of the proposed system. So it was improved the performance of SPARK with 38.53% over base system. Since it took only a few seconds (1356 sec). Fig:-10 represent the comparison between the proposed and base system's performance and table 3 refers proposed systems improved percentage over base system.

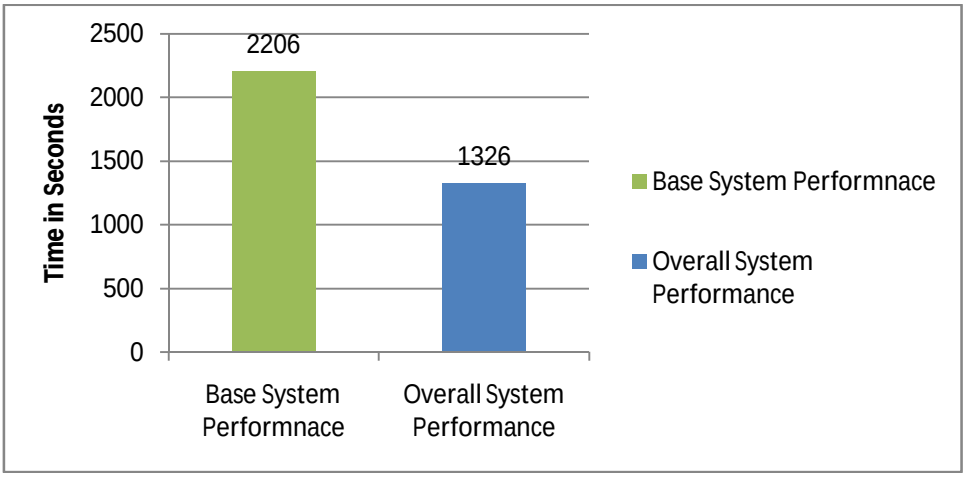

Fig.10.Comparison Chart between Base System and Proposed System3

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

Table III: THREE PROPSED SYETM& OVERALL SYSTEM PERFORMANE IN TERMS OF THE DURATION AND PERCENT IMPROVMENT

Name	Parameter	Time duration(in seconds)	Improvement(in percent)
Proposed System1	Impact by increasing reducer's container	1596	27.65
Proposed System2	Impact by increasing sort factor	1184	46.32
Proposed System3	Impact by increasing input buffer for shuffling	1526	30.82
Overall Proposed system	Impact by changing all three parameters	1356	38.53

VI. CONCLUSION AND FUTURE WORK

For this work, Twitter data was considered for analysis and explored on Apache SPARK default configuration parameter, known as base system. My research shows that modifying the parameters values for MapReduce job. To have better outcomes, configuration parameter must be adjust as per the specific application till awaiting resource fully used . SPARK admin should be cautious when you selecting and rectifying the parameters value. Since negligence leads degrade the performance. So in my paper-research, proposes the 'tuning approchment' optimizes the entire performance with 38.5% over default configuration SPARK framework. As of now so many hundred of parameters present at SPARK; thus I took only a few seconds for my research consideration. In my future work, I must take rest of the parameters those may also impact the inputs and outputs. And enhance SPARK performance as well. For furthers, a large data sets can be worked with a bigger cluster size, and some of the methods like Scheduling algorithms, resource management plans can help to enhance the SPARK's optimizing acute performance.

REFERENCES

1. Apache Spark. https://spark.apache.org
2. M.Hammoud, and M.F. Sakar," Locality-Aware Reduce Task scheduling for MapReduce" in Cloud Computing Technology and science (CloudCom),IEEE Third International Conference,2011, pp.570-576.
3. Z.tang, L.Jiang,J.Zhou, Kenli Li, and Keqin Li, " A self-adaptive scheduling algorithm for reduce start time" Future Generation Computer Systems, 2015, vol.43,pp.51-60.
4. R.Gu, X.Yang, J.Yan, Y.Sun, B.Wang, C.Yuan, and Y.Huang, "SHadoop: Improving MapReduce performance by optimizing job execution mechanism in Hadoop clusters", Journal of Parallel and Distributed Computing 74, 2014, vol. 3,pp.2166-2179.
5. H.Herodotou, H.Lim, G.Luo, N.Borisov, L.Dong, F.B.Cetin and S.babu, "Starfish: A Self-tuning System for Big Data Analytics" in Proceedings of the Fifth CIDR Conference, 2011, vol.11,pp.261-272.
6. H.Alashammari, J.Lee, and H.Bajwa, " Improving Hadoop Performance by Using Metadata of related jobs", IEEE Transactions on Cloud Computing, 2016.
7. Y.Guo,J.Rao, and X.Zhou, "iShuffle; improving Hadoop performance with shuffle-on-write" in proceedings of the 10th international Conference on Automatic Computing (ICAC'13).San Jose,CA,USENIX, 2013,pp.107-117.
8. K.Wang,X.Lin,and W.Tang. "Predator – An experience guided configuration optimizer for Hadoop MapReduce" in Cloud Computing Technology and Science (CloudCom), IEE 4th International Conference, 2012,pp. 419-426.
9. Prabhu, A.P.Rodrigues, G.Prasad, and H.R.nagesh, "Performance enhancement of Hadoop mapreduce framework for analyzing BigData" in Electrical, Computer and Communication Technologies (ICECCT), IEEE International Conference,2015,pp.1-8
10. V.N.Inukollu,S.Arsi and S.R.Ravuri, "Security issues associated with big data in cloud computing", International Journal of Network Security & Its Applications, 2014, vol.6,no.3, pp.42-56.
11. M.F.Uddin, and N.Gupta, "Seven V's of Big data understanding Big Data to extract value", in American Society for Engineering Eduction (ASEE Zone 1), 2014, pp.1-5.
12. I.Tomasic, J.Ugovsek, A.Rashkovska, and R.Trobec, "Multiuser Hadoop distributed file system" in MIPRO, proceedings of the 35th International Convention, 2012,pp.301-305.
13. V.Kalvri, and V.Vlassov, "MapReduce: Limitations, optimizations and open issues", in 12th IEEE International conference on trust, security and privacy in computing and communications, 2013, pp. 1031-1038.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 10, October 2018

14. S.Kaisler, F.Amour, J.A.Espinosa, and W.Money, "Big data: issues and challenges moving forward", in System Sciences(HICSS), 46th Hawaii international Conference,2013, pp.995-1004.
15. Shivath Babu, "Towards automatic optimization of MapReduce programs",in Proceedings of the 1st ACM symposium on cloud computing,2010,pp.137-142.
16. Intel, Optimizing Hadoop Deployments, Intel White Paper,2010.
17. AMD , Hadoop Performance Tuning Guide, AMD White Paper, 2012.
18. Tom White, Hadoop: The Definitive Guide, 4th ed., O'Reilly Media, Inc., 2015,pp. 185-208.[E-book].