Volume 5, Issue 9, September 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Big Data Analytics with Map Reduce and Hadoop

Dr. Rajender Bathla*
Department of CSA, HCTM
Haryana, India

Er. Nishi Midha
M.Tech Research Scholar
Haryana, India

Abstract— Recently, Big Data has attracted a lot of attention from academia, industry as well as government. It is a very challenging research area. Big Data is term defining collection of large and complex data sets that are difficult to process using conventional data processing tools. Every day, we create trillions of data all over the world. These data is coming from social networking sites, scientific experiments, mobile conversations, sensor networks and various other sources. We require new tools and techniques to organize, manage, store, process and analyse Big Data.Big Data analytics is the process of examining large amount of data i.e. Big Data and to uncover the hidden and unknown correlation to better understand the term. In analytics we analyse the tools used to process the Big Data. This paper represent Big data analytics using Hadoop and MapReduce tools using some of the basic concepts of these tools.

Keywords— Big Data,Hadoop, MapReduce, Algorithm, Terabytes, Streaming, Batch

I. INTRODUCTION

Big Data Analytics is the process of collecting, organising and analysing a large set of data to discover the pattern, logic and other useful information. Big data analytics help in understand the data and the information contained within the data and also help to identify the data that is most important to the business and future business decisions. Big Data Analysts basically want the knowledge of data that comes from analysing the data. [1]Henry kicked off a collaborative effort to examine some of the details behind the Big Data push and what they really mean. According to Henry "Big Data is the process of changing data into information, which then changes into knowledge". Big data analyticsis the use of advanced analytic techniques against very large, diverse data sets that include different types such as structured/unstructured and streaming/batch, and different sizes from terabytes to zettabytes. Big data is a term applied to data sets whose size or type is beyond the ability of traditional relational databases to capture, manage, and process the data with low-latency. And it has one or more of the following characteristics – high volume, high velocity, or high variety. Big data comes from sensors, devices, video/audio, networks, log files, transactional applications, web, and social media - much of it generated in real time and in a very large scale.[2]

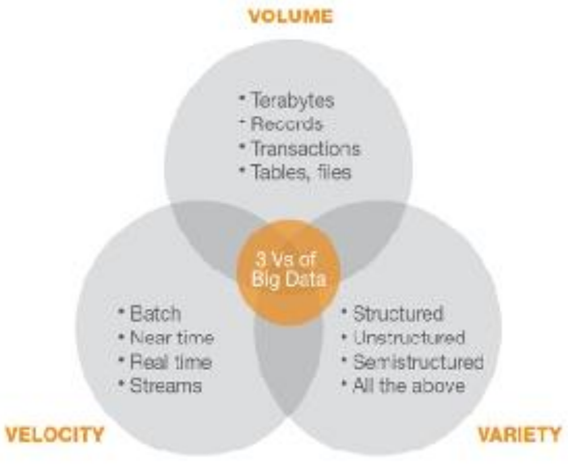

Fig 1: Characteristics of Big Data

Analyzing big data allows analysts, researchers, and business users to make better and faster decisions using data that was previously inaccessible or unusable. Using advanced analytics techniques such as text analytics, machine learning, predictive analytics, data mining, statistics, and natural language processing, businesses can analyze previously untapped data sources independent or together with their existing enterprise data to gain new insights resulting in significantly better and faster decisions. [1] There is a growing trend of applications that should handle big data. However, analyzing big data is a very challenging problem today. For such applications, the MapReduce framework has recently attracted a lot of attention. Google's MapReduce or its open-source equivalent Hadoop is a powerful tool for building such applications. In this paper, we will introduce the MapReduce framework based on Hadoop, discuss how to design efficient MapReduce algorithms.[3]

II. BIG DATA ANALYTICAL TOOLS

There are varieties of tools and techniques are developed by various organization to process and analyse Big Data. Big Data captured today is unstructured, from sensors used to gather climate information, posts on social media site, digital pictures and videos, purchase transaction records.[4] All unstructured data can be analyze through applications support parallelism with the help of computing clusters. The following are the tools in area of Big Data Analytics.

2.1 Hadoop

Hadoop cluster is a special type of computational cluster designed specifically for storing and analyzing huge amount of structured and unstructured data in a distributed computing environment.[4]Hadoop is a large scale, open-source software framework that supports a large data sets in distributed computing environment. Distributed analytic frameworks, such as MapReduce, are evolving into distributed resource managers that are gradually turning Hadoop into a general-purpose data operating system, says Hopkins.[5] With these systems, he says, "you can perform many different data manipulations and analytics operations by plugging them into Hadoop as the distributed file storage system." [3]Hadoop is dedicated to scalable, distributed, data-intensive computing. Hadoop used to handle thousands of petabytes of data in different clusters. Hadoop is the core platform for structuring Big Data, and solves the problem of formatting it for subsequent analytics purposes.[6]Hadoop uses a distributed computing architecture consisting of multiple servers using commodity hardware, making it relatively inexpensive to scale and support extremely large data stores. Apache Hadoop, a nine-year-old open- source data-processing platform first used by internet gaints including Yahoo and Facebook, leads the big –data revolution. Cloudera introduced a commercial support for expertise in 2008, and MapR and Hortonworks pilled on in 2009 and 2011, respectively.[7] Clusters runsHadoop's Open source distributed processing software on low-cost commodity computers. Typically one machine in the cluster is designated as the NameNode and another machine the Job Tracker; these are the masters. The rest of the machines in the cluster act as both DataNodes and TaskTracker; these are the slaves. Hadoop clusters are often referred to as "shared nothing". Hadoop clusters are known for boosting the speed of data analysis applications. If a cluster's processing power is overwhelmed by growing volumes of data, additional cluster nodes can be added to increase throughout.[8]Hadoop clusters also are highly resistance to failure because each piece of data is copied onto other cluster nodes, which ensures that the data is not lost if one node fails. As of early 2013, Facebook was recognized as having the largest Hadoop cluster in the world. Other prominent user includes Google, Yahoo and IBM.[4][9]

2.1.1 Hadoop Architecture: At a high-level, Hadoop operates on the philosophy of pushing analysis code close to the data it is intended to analyze rather than requiring code to read data across a network. As such, Hadoop provides its own file system, named as*Hadoop File System* or *HDFS*. When you upload your data to the HDFS, Hadoop will partition your data across the cluster (keeping multiple copies of it in case your hardware fails), and then it can deploy your code to the machine that contains the data upon which it is intended to operate.[10]

HDFS organizes data by keys and values. Each piece of data has a unique key and a value associated with that key.[8] Relationships between keys, if they exist, are defined in the application, not by HDFS. And in practice, we analyze a problem domain in order realize the full power of Hadoop (see the next section on MapReduce).[11]

The components that comprise Hadoop are:

2.1.2 HDFS: The Hadoop file system is a distributed file system designed to hold huge amounts of data across multiple nodes in a cluster.Hadoop provides both an API and a command-line interface to interacting with HDFS.

MapReduce Application:

The next details of MapReduce, but in short, MapReduce is a functional programming paradigm for analyzing a single record in your HDFS. It then assembles the results into a consumable solution. The Mapper is responsible for the data processing step, while the Reducer receives the output from the Mappers and sorts the data that applies to the same key.

Partitioner: The partitioner is responsible for dividing a particular analysis problem into workable chunks of data for use by the various Mappers. The HashPartioner is a partitioner that divides work up by "rows" of data in the HDFS, but you are free to create your own custom partitioner if you need to divide your data up differently.

Combiner: If, for some reason, you want to perform a local reduce that combines data before sending it back to Hadoop, then you'll need to create a combiner. A combiner performs the reduce step, which groups values together with their keys, but on a single node before returning the key/value pairs to Hadoop for proper reduction.

InputFormat: Most of the time the default readers will work fine, but if your data is not formatted in a standard way, such as "key, value" or "key [tab] value", then you will need to create a custom InputFormat implementation.

Output Format: MapReduce applications will read data in some InputFormat and then write data out through an OutputFormat. Standard formats, such as "key [tab] value", are supported out of the box, but if you want to do something else, then you need to create your own OutputFormat implementation.

Additionally, Hadoop applications are deployed to an infrastructure thatsupports its high level of scalability and resilience.[12]These components include.

NameNode: The NameNode is the master of the HDFS that controls slave DataNode daemons; it understands where all of your data is stored, how the data is broken into blocks, what nodes those blocks are deployed to, and the overall health of the distributed filesystem.

- **Secondary NameNode**: The Secondary NameNode monitors the state of the HDFS cluster and takes "snapshots" of the data contained in the NameNode. If the NameNode fails, then the Secondary NameNode can be used in place of the NameNode. This does require human intervention, however, so there is no automatic failover from the NameNode to the Secondary NameNode, but having the Secondary NameNode will help ensure that data loss is minimal. Like the Name Node, each cluster has a single Secondary Name Node.

- **Job Tracker**: The Job Tracker daemon is your liaison between your application and Hadoop itself. There is one Job Tracker configured per Hadoop cluster and, when you submit your code to be executed on the Hadoop cluster, it is the Job Tracker's responsibility to build an execution plan. This execution plan includes determining the nodes that contain data to operate on, arranging nodes to correspond with data, monitoring running tasks, and relaunching tasks if they fail.

Task Tracker: Similar to how data storage follows the master/slave architecture, code execution also follows the master/slave architecture. Each slave node will have a Task Tracker daemon that is responsible for executing the tasks sent to it by the Job Tracker and communicating the status of the job with the Job Tracker.

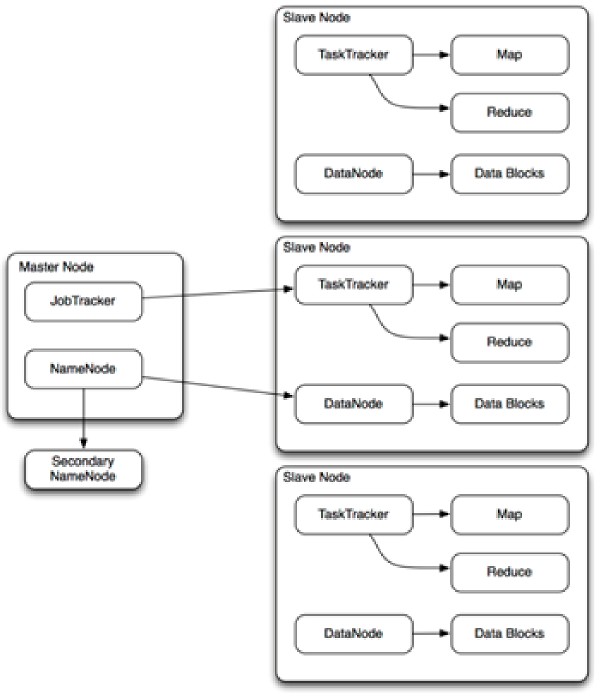

Fig 2: Hadoop application and Infrastructure interactions.

Figure 2 shows the relationships between the master node and the slave nodes. The master node contains two important components: the Name Node, which manages the cluster and is in charge of all data, and the Job Tracker, which manages the code to be executed and all of the Task Tracker daemons. Each slave node has both a Task Tracker daemon as well as a Data Node: the Task Tracker receives its instructions from the Job Tracker and executes map and reduce processes, while the Data Node receives its data from the Name Node and manages the data contained on the slave node. And of course there is a Secondary Name Node listening to updates from the Name Node.[5][13] Hadoop recently became very popular with several different vendors offering distributions with a set of optimizations and features. Datameer is committed to supporting all of the Hadoop distributions and allows easy migration from one to other. Datameer isolates the end user from the lower level technical details and provides an simple though powerful web bases application on top that abstracts all interactions with Hadoop. Some of distributors preferingHadoop platforms are: Apache, Amazon, Cloudera, EMC, Hortonworks, IBM, MapR, Microsoft etc.

2.2 MapReduce

MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster.[14]A MapReduce program is composed of a Map () procedure that performs filtering and sorting and a Reduce () procedure that performs a summary operation. The "MapReduce System" manages the processing by marshalling the distributed servers, running the various tasks in parallel, managing all communications and data transfers between the various parts of the system, and providing for redundancy and fault tolerance. The model is inspired by the map and reduce functions commonly used in functional programming,although their purpose in the MapReduce framework is not the same as in their original forms. The key contributions of the MapReduce framework are not the actual map and reduce functions, but the scalability and fault-tolerance achieved for a variety of applications by optimizing the execution engine once.[15] As such, a single-threaded implementation of MapReduce (such as MongoDB) will usually not be faster than a traditional (non-MapReduce) implementation, any gains are usually only seen with multi-threaded implementations. The use of this model is beneficial only when the optimized distributed shuffle operation (which reduces network communication cost) and fault tolerance play. Optimizing the communication cost is essential to a good MapReduce algorithm.[16]Google inventedMapReduceto deal the issues of how to parallelize the computation, distribute the data, and handle failures.

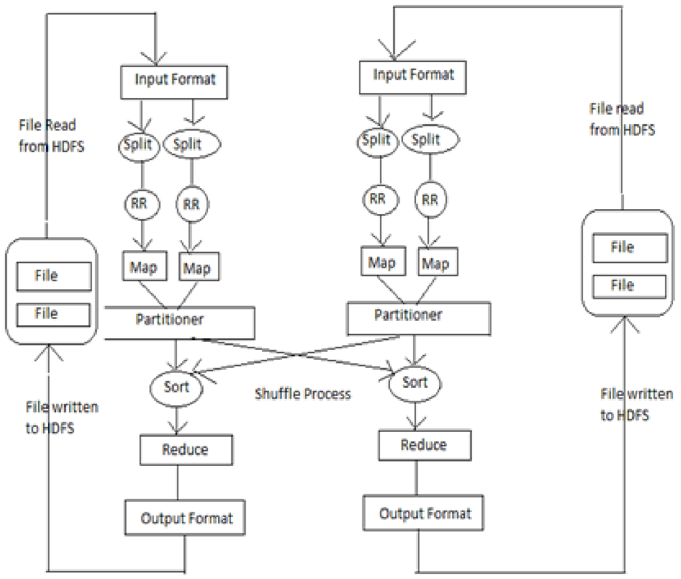

Fig 3: Flow Chart for MapReduce

2.2.1.Prepare the Map() input

The system splits the input files into M pieces and then starts up M Map workers on a cluster of machines.

2.2.2.Run the user-defined Map() code

The Map worker parses key-value pairs out of the assigned split and passes each pair to the user- defined Map function. The intermediate key-value pairs produced by the Map function are buffered in memory. Periodically, the buffered pairs are written to local disk, partitioned into R regions for shading purposes by the partitioning function that is given the key and the number of reducers R and returns the index of the desired reducer.

2.2.3.Shuffle the Map output to the Reduce processors

When ready, a reduce worker reads remotely the buffered data from the local disks of the map workers. When a reduce worker has read all intermediate data, it sorts the data by the intermediate keys so that all occurrences of the same key are grouped together. Typically many different keys map to the same reduce task.

2.2.4.Run the user-defined Reduce () code

The reduce worker iterates over the sorted intermediate data and for each unique intermediate key encountered, it passes the key and the corresponding set of intermediate values to the user's Reduce function.

2.2.5.Produce the final output

The final output is available in the R output files (one per reduce task).

Optionlly, a combiner can be used between map and reduce as an optimization. The combiner function runs on the output of the map phase and is used as a filtering or an aggregating step to lessen the data that are being passed to the reducer. In most of the cases the reducer class is set to be the combiner class so that we can save network time.[17] MapReduce provides programmers a simple parallel computing paradigm. Inspired by functional programming, the computing paradigm of MapReduce is really simple and easy to understand. Because of automatic parallelization, no explicit handling of data transfer and synchronization in programs, and no deadlock. This model is very attractive. MapReduce is also designed to process very large data that is too big to fit into the memory.[18] To achieve that, MapReduce employs a data flow model, which also provides a simple I/O interface to access large amount of data in distributed file system. It also exploits data locality for efficiency. In most cases, we don't need to worry about I/O at all.Now let's look into several examples, which will also help us understand the limitations of MapReduce.

Example 1: Sort

The essential part of the MapReduce framework is a large distributed sort. So we just let the framework do the job in this case while the map is as simple as emitting the sort key and original input. The reduce operator is an identity function.

```
Public class SortingThread
Sorting(ArrayList al, int fromIndex, int toIndex, intfno)
 If(loc>begin+1)
Sorting stL=new Sorting (arl,begin,loc-1,fno)
Sorting left start
If (loc<end-1)
SortingThread stR=new SortingThread (arl,loc+1,end,fno)
Sorting right start
Public void run()
Set left=begin
Set right=end
While(left<right)
```

```
13. Record left=array list(left)
14. Record right= array list(right)
15. Value left= Record[fno]
16. Value right= Record[fno]
17. If (valR>0)
18. Right—
19. Else If(left==right)
20. Set Loc=left
21. Split(left)
22. End If
23. Set temp1=arl(left)
24. Set temp2=arl(right)
25. Swap(left,temp2)
26. Swap(right,temp1)
27. While(right>left)
28. Recl=arl.get(left)
29. Rec2=arl.get(right)
30. valueL=recL[fno]
31. valueR=recR[fno]
32. If(valL>0)
33. Break
34. Else left++
35. End while
36. If(left==right)
37. Loc=left
38. Temp1=arl(left)
39. Temp2=arl(right)
40. Swap arl(left,temp2)
41. Swap arl(right,temp1)
42. End run
43. End class
```

Example 2: Join

A join combines records from two or more data sets by a common key. There are several ways to implement join in MapReduce. A straightforward approach is the reduce-side joins that take advantage of the identical keys to the same reducer. In practice, join, aggregation, and sort are frequently used together, e.g. finding the student scores maximum during the period. In MapReduce, this has to be done in multiple phases. The first phrase filters the data base on the click timestamp and joins the client and click log datasets. The second phrase does the aggregation on the output of join and the third one finishes the task by sorting the output of aggregation.

```
1. Public class JoinThread
2. JoinThread(ArrayList r1,ArrayList r2,ArrayList r3,int r)
3. Public void run()
4. String A[]=rsl1.get(rno)
5. String B[]=rsl2.get(rno)
6. String C[]=rsl3.get(rno)
7. For(i=1;i<=4;i++)
8. A=parse integer(A[i])
9. B=parse integer(B[i])
10. C=parse integer(C[i])
11. Total[i-1]=a+b+c
12. End for
13. End run
14. End class
```

Example 3: Aggregation

Aggregation is a simple analytic calculation such as counting the number of access or users from different colleges. Word count, the hello world program in the world of MapReduce, is an example of aggregation.

```
1. Public class FrequencyThread
2. FrequencyThread(String nm,ArrayList al1,ArrayList al2)
3. Public void run()
4. If (alA.contains(name))
5. Int i= alA.indexOf(name)
6. Int f= Integer parse int((String)alB.get(i))
```

```
7.  f++
8.  set alB(I,f+"")
9.  Else
10. alA.add(name)
11. Add 1 to alB
12. End if
13. End class
```

MapReduce is useful for batch processing on terabytes or petabytes of data stored in Apache Hadoop.

III. SCOPE IN REAL-WORLD FOR BIG DATA ANALYTICS

With data growing so rapidly and the rise of unstructured data accounting for 90% of the data today, the time has come for enterprises to re-evaluate their approach to data storage, management and analytics.[19] Legacy systems will remain necessary for specific high-value, low-volume workloads, and complement the use of Hadoop -optimizing the data management structure in your organization by putting the right system.
The cost effectiveness, scalability, and streamlined architectures of Hadoop will make the technology more and more attractive.
Here are some real-world examples of Big Data in action:

- Consumer product companies and retail organizations are monitoring social media like Facebook and Twitter to get an unprecedented view into customer behavior, preferences, and product perception.
- Manufacturers are monitoring minute vibration data from their equipment, which changes slightly as it wears down, to predict the optimal time to replace or maintain. Replacing it too soon wastes money; replacing it too late triggers an expensive work stoppage
- Manufacturers are also monitoring social networks, but with a different goal than marketers: They are using it to detect aftermarket support issues before a warranty failure becomes publicly detrimental.
- The government is making data public at both the national, state, and city level for users to develop new applications that can generate public good.

IV. LIMITATION OF HADOOP/MAPREDUCE

4.1 Cannot control the order in which the maps or reductions are run.
4.2 For maximum parallelism, you need Maps and Reduces to not depend on data generated in the same MapReduce job (i.e. stateless)
4.3 A database with an index will always be faster than a MapReduce job on unindexed data
4.4 Reduce operations do not take place until all Maps are complete (or have failed then been skipped)
4.5 General assumption that the output of Reduce is smaller than the input to Map; large data source used to generate smaller final values

V. CONCLUSION

This paper is a systematic study of various tools for Big data analytics. Big data is very challenging research area. Data is too big to process using conventional tool of data processing. Academia and industry has work together to design and develop new tools and technologies which effectively handle to processing of Big Data. Big Data is an emerging trend and there is immediate need for new machine learning and data mining techniques toanalyze massive amount of data in future. After years of practice, the community has realized these problems and try to address them in different ways. In this paper we conclude Hadoop&MapReduce tools. These tools are used for storing of structured\unstructured data in large amount this can be done through HDFS in which data is stored in clusters. MapReduce is used to process such a large amount of data in systematic manner. By using HadoopMapReduce algorithms we can access such a large amount of data in very efficient time.We try to simulate a university system by using the basic algorithms of Hadoop and MapReduce. We can also process this data with the queries or through the interactive frames. We prefer to use the data processing through interactive frame work in which we can access the result of students from different colleges. And can also capable of getting all the records from clusters with high efficiency. Enterprises are increasingly looking to find actionable insights into their data. Many big data projects originate from the need to answer specific business questions. With the right big data analytics platforms in place, an enterprise can boost sales, increase efficiency, and improve operations, customer service and risk management. These tools are used to boost the level of the processing time of accessing and processing in efficient time.

REFERENCE

[1] A, Katal, Wazid M, and Goudar R.H. "Bid data: Issues, challenges, tools and Good practices" Noids:2013,pp. 404-409,8-10 Aug. 2013.-
[2] Jimmy Lin, Chris Dyer," Data-Intensive Text Processing with Map Reduce", Manuscript prepared April 11, 2010.
[3] Tom White," Hadoop: The Definitive Guide", O'Reilly Media, Inc., 2009.
[4] Lu, Huang, Ting-tin Hu, and Hai-shanChen,"Research on Hadoop cloud computing Model and its applications" Hangzhou, china: 2012, pp.59-63, 21-24 Oct.2012.

[5] Geczy, P., Izumi,N.,&Hasida, K. (2012). Cloudsourcing: Managing cloud adoption. Global Journal of Business Research, 6(2), 57-70
[6] Cole, B. (2012). Looking at business size, budget when choosing between SaaS and hosted ERP. E-guide: Evaluating SaaS vs. on premise for ERP - systems. Retrieved from http://docs.media.bitpipe.com/io_10x/io_104515/item_548729/SAP_sManERP_IO%23104515_EGuide_061212.pdf
[7] ClouderaMapReduce Algorithms, 2009 Cloudera, inc.
[8] http://searchcloudcomputing.techtarget.com/defination/Infrastructure-as-a-Service-IaaS
[9] http://www- 01.ibm.com/software/data/infosphere/hadoop/what-is-big-data-analytics.html
[10] https://en.wikipedia.org/wiki/Big_data
[11] http://www.computerworld.com/article/2690856/big-data/8-big-trends-in-big-data-analytics.html
[12] http://searchbusinessanalytics.techtarget.com/definition/Hadoop-cluster
[13] http://www.informit.com/articles/article.aspx?p=2008905
[14] https://en.wikipedia.org/wiki/Big_data
[15] http://www.technologytransfer.eu/article/98/2012/1/What_Is_Big_Data_and_Why_Do_We_Need_It_.html
[16] http://asmarterplanet.com/blog/2014/10/big-data-analytics-caring.htm
[17] http:// arunxjacob.blogspot.in/2011/04/hdfs-file-size-vs-allocation-other.htm
[18] Apache Giraph Project, http://giraph.apache.org/
[19] http://www.webopedia.com/big_data_analytics.html
[20] http://searchbusinessanalytics.techtarget.com/essentialguide/Guide-to-big-data-analytics-tools-trends-and-best-practices

ABOUT AUTHOR

Dr. Rajender Kumar Bathla, author of the present paper is working as an Assistant Professor in Computer Science & Engineering Department at HCTM Technical Campus, Kaithal. He started his professional career from Haryana College of Technology & Management, Kaithal in 2004 just after graduating his MCA Degree from MDU Rohtak and later he accomplished M.Tech. Degree with specialization in Information Technology from M.M. University Mullana, Haryana in 2009 and earned Ph.D. Degree also in 2012 in the Discipline of Computer Science & Engineering, Faculty of Technology from Private State University, (India) under supervision and guidance of his Senior Colleague & Advisor Prof. (Dr.) V. N. Maurya, Former Founder Director, Vision Institute of Technology Aligarh, U.P. Technical University, India. His research areas are mainly Software Testing and Data Structures and published several research papers in Indian and Foreign journals and Conferences. **Dr. Rajender Kumar Bathla** is an associated with several research and profession bodies**, IEEE, ISTE, CSTA, UACEE, IACSIT, IIST, IAEST,IAENG and ISA**. Apart from this, he is also on role of Editor and Reviewer of several Foreign leading International journals such as of **IJSEA USA', 'IJOAIEST-INDIA', 'IJECSE- TAIWAN', 'IJOAR&T –USA.**