ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 12, December 2014

Extract Transform and Load Strategy for Unstructured Data into Data Warehouse Using Map Reduce Paradigm and Big Data Analytics

P.Saravana kumar[1], M.Athigopal[2], S.Vetrivel[3]

Assistant Professor, Dept of Computer Application, Senthamarai College of Arts and Science, Tamilnadu, India[1]

Assistant Professor, Vice Principal, Dept of Information Technology and Networking, Subbalakshmi Lakshmipathy College of Science, Tamilnadu, India[2, 3]

ABSTRACT: Analytics over the huge volume of data is now possible with Big data. Data keep on accumulated on every minute from multitude data sources such as social media, mobile devices, and sensors. In order to extract insights from diverse information feeds from multiple, often unrelated sources, data need to be correlated or harmonized to a common level of granularity. Loading Unstructured Data into Data warehouse getting complex. A strategy for fetching the unstructured data into Hadoop Distributed File System is discussed. Data cleansing and profiling of extracted data is important to overcome data quality concerns. Transform phase carried with map reduce frame work. Computation ratio, Network band width and Data locality parameters are monitored with full dump and Incremental load operations. Pig Latin is used to process data from Hadoop Distributed File System and finally load the process data into HDFS file or Data warehouse. Aggregated data from Pig is minimal Subset of Data is Loaded to Data warehouse for Business Analytics and Enterprise Reporting. Based on the Performance related parameters appropriate strategy is suggested for Different type of application.

KEYWORDS: Big data Analytics, Data warehouse, Map-reduce Paradigm, ETL Process.

I. INTRODUCTION

In data warehousing, ETL flows are responsible for collecting data from different data sources, transformation, and cleansing to comply with user-defined business rules and requirements. Current ETL technologies are demanded to process many gigabytes of data each day. The vast amount of data makes ETL extremely time-consuming. The use of parallelization technologies is the key to achieve better ETL scalability and performance. In recent years, the "cloud computing" technology MapReduce has been widely used for parallel computing in data-intensive areas due to its good scalability.

We see that Map Reduce can be a good foundation for ETL parallelization. ETL processing exhibits the composable property such that the processing of dimensions or facts can be split into smaller computations and the partial results from these computations can be merged to constitute the final results in a DW. Further, the MapReduce programming paradigm is very powerful and flexible.

MapReduce makes it easier to write a distributed computing program by providing interprocess communication, fault-tolerance, load balancing, and task scheduling. It is often said that while parallel DBMSs are good for querying of large data sets, MapReduce- based systems are good for ETL tasks and that a MapReduce-based ETL system thus can live upstream from the DBMS. However, MapReduce is a general framework and lacks support for high- level ETL-specific constructs such as star and snowflake schemas.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 12, December 2014

II. RELATED WORK

In [2] authors Analysing the process of map tasks in Hadoop, the serial execution of data transmission and data processing is discovered to cause overhead when the input data is not local. The proposed data prefetching mechanism overlap data transmission with data processing Parameters including map processing time, data transmission time, total input data size and input split size of map tasks in experimental applications are also varied to simulate different conditions. The experiment results show that up to 94% data transmission time is reduced and up to 15% performance improvement in jobs' execution is achieved with the proposed mechanism [3]. Size of data grows; traditional distributed log data processing systems are not able to processing massive log data from different applications with millions of users. This paper proposes a mass log data processing and data mining methods based on Hadoop to achieve scalability and performance. The model, process, architecture, and implementation of the data processing and mining methods are proposed, and the experimental results is shown and analysed to prove the effectiveness of the methods.Analysing the process of map tasks in Hadoop, the serial execution of data transmission and data processing is discovered to cause overhead when the input data is not local. The proposed data prefetching mechanism overlap data transmission with data processing Parameters including map processing time, data transmission time, total input data size and input split size of map tasks in experimental applications are also varied to simulate different conditions. The experiment results show that up to 94% data transmission time is reduced and up to 15% performance improvement in jobs' execution is achieved with the proposed mechanism [4].Size of data grows; traditional distributed log data processing systems are not able to processing massive log data from different applications with millions of users. This paper proposes a mass log data processing and data mining methods based on Hadoop to achieve scalability and performance. The model, process, architecture, and implementation of the data processing and mining methods are proposed, and the experimental results is shown and analysed to prove the effectiveness of the methods [5].

III. PROPOSED SYSTEM

A. Design Considerations:

- Loading Unstructured Data into Data warehouse getting complex. Strategies for fetching the unstructured data into Hadoop Distributed File System is discussed.
- Data cleansing and profiling of extracted data is important to overcome data quality concerns. Transform phase carried with map reduce frame work. Computation ratio, Network band width and Data locality parameters are monitored with full dump and Incremental load operations.
- Pig Latin is used to process data from Hadoop Distributed File System and finally load the process data into HDFS file or Data warehouse.
- Aggregated data from Pig is minimal Subset of Data is Loaded to Data warehouse for Business Analytics and Enterprise Reporting.
- Based on the Performance related parameters appropriate strategy is suggested for Different type of batch Loads in ETL

B. Approaches used in the proposed system:

Application Server generally runs on thousands of servers to accommodate millions of users. The log data is generated at the speed of millions of pieces per second, and the overall amount can reach hundreds of billions. The average size of one piece of log data is hundreds bytes, so the overall log data size can reach the level of PB (peta-bytes). Current log data processing systems based on RDBMS cannot cope with problems with such massive data.

Although there is optimization method for RDBMSs with big data based on index optimization and distributed technologies, RDBMSs do have limits when facing data beyond certain level. Under the circumstances where data size is bigger than TB level, the performance of RDBMSs deceases sharply, which makes log data processing based on RDBMSs ineffective, in some cases even unavailable to provide real time statistics. Hadoop is a framework designed for processing big data on large cluster built of commodity hardware. The Hadoop provides both reliability and efficiency. Hadoop contains a computational paradigm named Map Reduce, where the application is divided into many small fragments of work, each of which may be executed or re-executed on any node in the cluster.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 12, December 2014

In addition, it provides a distributed file system (HDFS) that stores data on the compute nodes, providing very high aggregate bandwidth across the cluster. Both Map Reduce and HDFS are designed so that node failures are automatically handled by the framework. Based on Map Reduce and HDFS, Hadoop is suited for solve problems of log data processing.

C. Detail Analysis on Application server Architecture:

In log data processing, Application server runs on clusters. The log data of execution information is generated during the execution phase, then transmitted and stored through queue asynchronously.

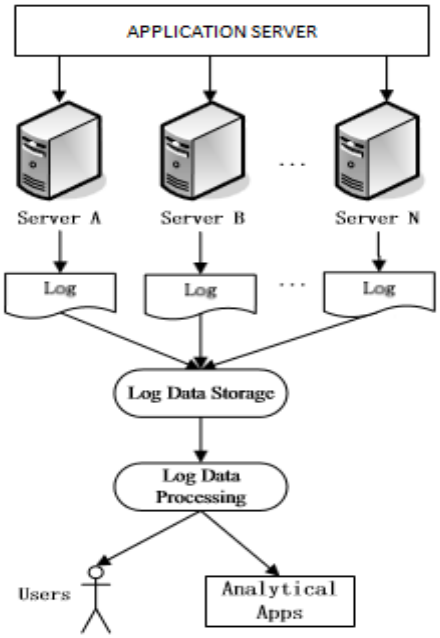

Fig.1 Log File Processing

Fig.1 shows the stored log data is processed according to specific requirements. Users want to see the usage and billing information while analytical applications need real time statistics of online users, responding time, and etc. To be able to provide the above mentioned statistics, the log should contain information of time, tenant, user, functions and resources. The execution instance, which means each time of service execution invoked by users, generates one or several pieces of logs.

D. Need for HDFS and Map-Reduce Paradigm:

- Normal Log file processing involves processing the log file into reduced data set to load the processed data set in to database.
- Computation resources for Normal Log file processing is less
- Cluster size keeps on increasing with application demand
- Size of Log File Growing exponentially
- Log Files of More than 2 TB of Size getting generated on daily basis. These log files very hard to process in single machine
- These files are loaded into HDFS (Hadoop Distributed File System)
- HDFS boon for Large files storing and Large file processing

Map-Reduce Programming Patterns Used to process the distributed files information.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 12, December 2014

E .HDFS Architecture:

- Fig.2 shows the HDFS (Hadoop Distributed File System) is Lower Granular level to store the Information
- Name Node and Data Node have HDFS Components. Users Data available in the HDFS of Data nodes. Name Node Controls the Mapping where the Information gets Stored. Name Node Coordinates the Tasks to Data nodes by task tracker and each data node have task tracker to get the tasks for further computation (Sort,Merge,Join,Aggregation)
- Task tracker in Data Nodes sends the Information to the Job tracker in the Name Node via Ethernet Switch or Separate LAN.Performance of the Data movement depends on the Network Bandwidth and Switch Speed
- For High Availability of Information, Data gets stored in data nodes is replicated. Based on the replication parameter in Configfile.xml in Hadoop distribution Job Trackers have Scheduling Algorithms for Scheduling the Map-reduce Tasks. Over Direct MR Tasks or Higher level of abstraction tasks PIG Jobs or Query in HIVE

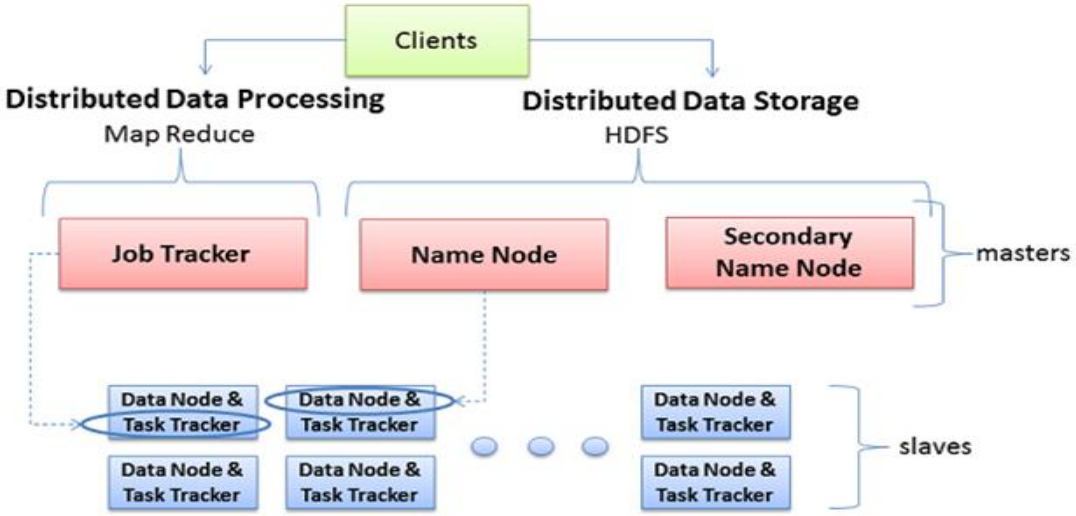

Fig. 2 HDFS Functional Overview

E. Map – Reduce:

Fig. 3 shows Map Reduce Framework classified into Two Parts:

- Map Task

1. Map Function
2. Sort
3. Shuffle

- Reduce Task
  1. Merger
  2. Reduce Function
  3. Output

Map Tasks Follows the respective phases. Map Function maps the Sequential log files into HDFS data files across Nodes and Sort phase sorts the data in key value pairs. Key value based on any filed or line number from the file. Shuffle phase Intermediate phase between Map and reduce phase its Joins the all Sorted datasets to each other and Forms the resultant data set. It's most time consuming stage. Merger Stage helps to collate the appropriate dataset .Reduce Function most aggregated stage based on business logic we applied. Finally output phase carry over the reduced datasets to HDFS, External sequential file or Database itself using Sqoop.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 12, December 2014

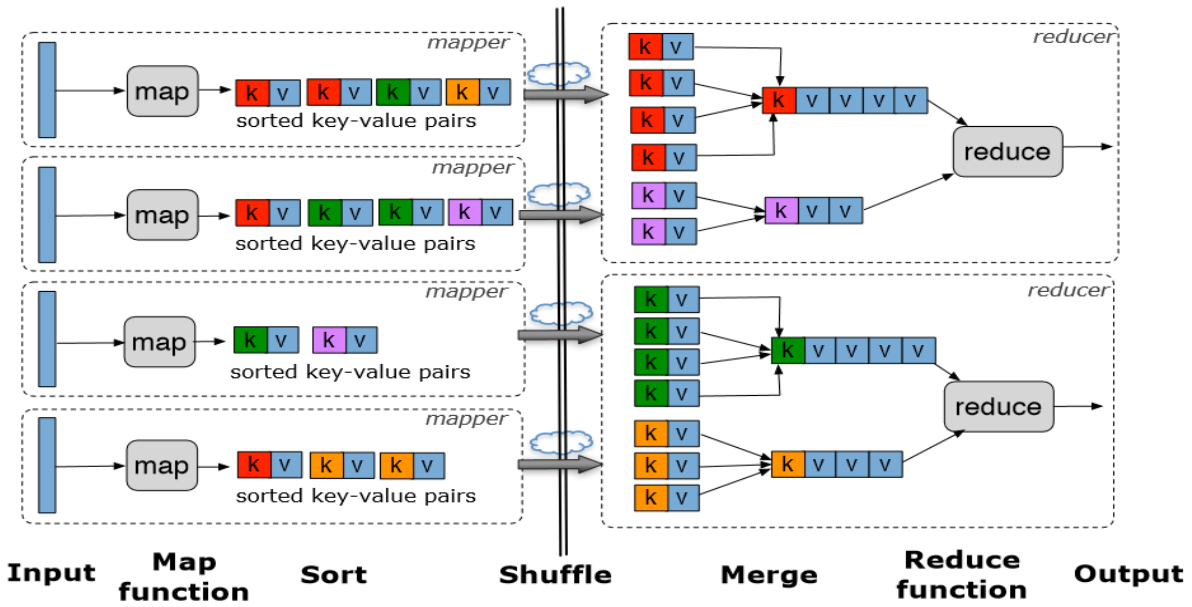

Fig. 3. Map Reduce Data Flow Diagram

III. SIMULATION RESULTS

```
biadmin@linux-eqlj:~/PROJECT/SCRIPT_FILES> ./UNSTRUCTURED_TO_DW.sh
--- COPY THE DATA TO HDFS -  .... In Progress
--- COPY THE DATA TO HDFS -  .... Completed
--- Triggering the Map-Reduce Job
--- In Progress 10 % ..
--- In Progress 20 % ...
--- In Progress 30 % ....
--- In Progress 40 % .....
--- In Progress 50 % ......
--- In Progress 60 % .......
--- In Progress 70 % ........
--- In Progress 80 % .........
--- In Progress 90 % ..........
--- In Progress 100 % ..........
 .. Job Completed ..
biadmin@linux-eqlj:~/PROJECT/SCRIPT_FILES>
```

Fig. 4 Unix Shell Script For Reading the Log File

Fig.4 shows As of Two approaches Pull method and Push method with Quality Addressing the Second approach improves the Important Performance related Parameters such as Computation ratio, Network bandwidth and Data Locality Factors get improved by 6 % of its Respective Metrics

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 12, December 2014

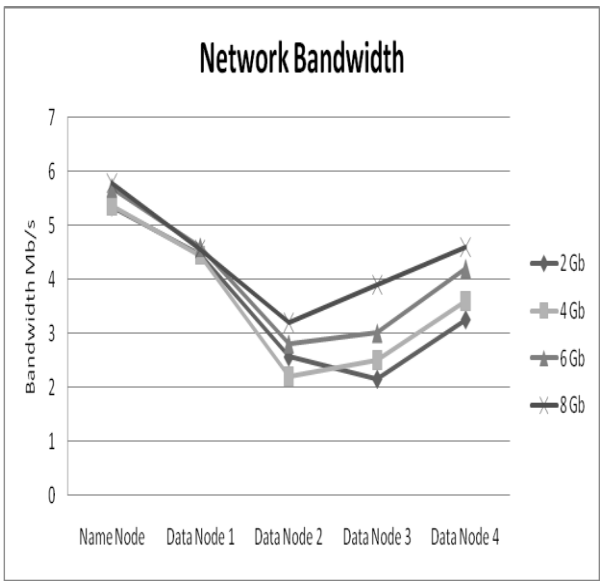

Fig.5 Computation Ratio across Nodes

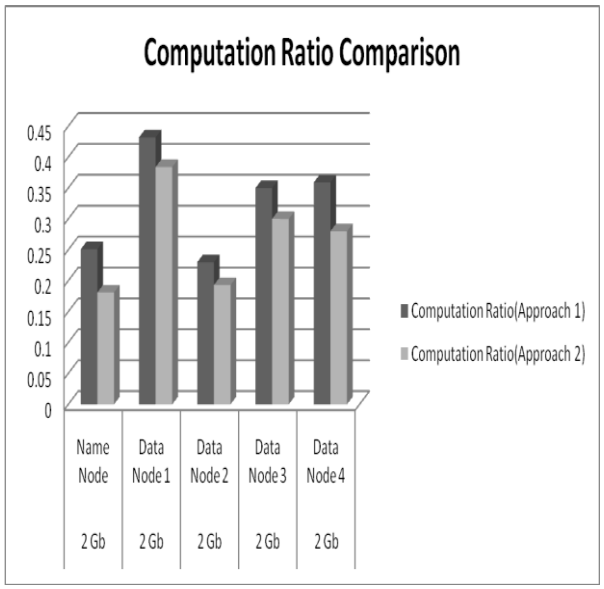

Fig. 6. Network Bandwidth across Nodes

Fig.5 and 6 shows the ratio in the shuffle method, the intermediate data generated by the map phase is fetched. Each reduce task is assigned a partition of the intermediate data with a fixed the key range, so the reduce task must fetch the content of this partition from every map task's output in the cluster.

Data set Size	Name Node	Data Node 1	Data Node 2	Data Node 3	Data Node 4
2 Gb	2	34	44	20	0
4 Gb	4	28	35	0	33
6 Gb	9	0	45	30	14
8 Gb	12	25	0	23	40

Table 1. Data File Distribution across HDFS

Table.1 shows Task tracker in Data Nodes sends the Information to the Job tracker in the Name Node via Ethernet Switch or Separate LAN.Performance of the Data movement depends on the Network Bandwidth and Switch Speed.

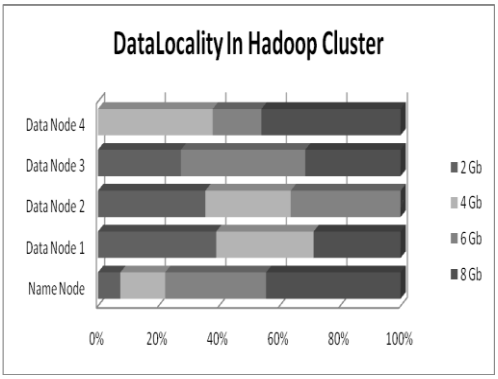

Fig. 7. Data Locality Factor Across Hadoop Nodes

Map-Reduce Framework	Virtual memory (bytes) snapshot	2,425,442,304	912,453,632	3,337,895,936
	Reduce input groups	0	9,999,000	9,999,000
	Combine output records	19,998,000	0	19,998,000
	Map output records	9,999,000	0	9,999,000
	CPU time spent (ms)	478,190	291,020	769,210
	Map input records	9,999,000	0	9,999,000
	Reduce shuffle bytes	0	463,814,609	463,814,609
	Combine input records	19,998,000	0	19,998,000
	Spilled Records	19,998,000	9,999,000	29,997,000
	SPLIT_RAW_BYTES	2,418	0	2,418
	Map output bytes	526,643,461	0	526,643,461
	Reduce input records	0	9,999,000	9,999,000
	Physical memory (bytes) snapshot	1,386,729,472	251,342,848	1,638,072,320
	Total committed heap usage (bytes)	811,541,504	20,969,472	832,510,976
	Reduce output records	0	9,999,000	9,999,000
	Map output materialized bytes	546,641,533	0	546,641,533

Fig. 8. Performance Related Parameters in Map reduce Tasks

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 12, December 2014

Fig.7 and 8 shows the task then enters the last reduce step, in which the user-defined reduce function is invoked for each distinct key in a sorted order, passing it the associated list of values. The output of the reduce function is written to a temporary location on HDFS. After the reduce function has been applied to each key in the reduce task's partition, the task's HDFS output file is automatically renamed from its temporary location to its final location.

IV. CONCLUSION AND FUTURE WORK

Loading Unstructured Data into Data warehouse getting complex. Strategies for fetching the unstructured data into Hadoop Distributed File System is discussed. Data cleansing and profiling of extracted data is important to overcome data quality concerns. Transform phase carried with map reduce frame work. Computation ratio, Network band width and Data locality parameters are monitored with full dump and Incremental load operations. Pig Latin is used to process data from Hadoop Distributed File System and finally load the process data into HDFS file or Data warehouse. Aggregated data from Pig is minimal Subset of Data is Loaded to Data warehouse for Business Analytics and Enterprise Reporting. Based on the Performance related parameters Second Approach Push method with Quality Addressing will be best one for Suggested ETL batch application.

REFERENCES

1. Puneet Agarwal, Gautam Shroff, Pankaj Malhotra, "Approximate Incremental Big-DataHarmonization", IEEE Transaction on Bigdata Congress, Vol 5, Issue No 2, June 2013.
2. Hongyong Yu, Deshuai Wang Proc., " Mass Log Data Processing and Mining Based on Hadoop and Cloud Computing ", Computer Science & Education (ICCSE 2012)July 14-17, 2012.Melbourne, Australia
3. Guozhang Wang, Marcos Vaz Salles, Benjamin Sowell, Xun Wang, Tuan Cao, Alan Demers, Johannes Gehrke, "Behavioral Simulations in Map Reduce", Walker White Procedings of VLDB endorsement Vol 3 No 1 2012
4. Aysan Rasooli Oskooei, "Improving Scheduling in Heterogeneous Grid and Hadoop Systems", Open Access Desertions and Theses,May 2013.
5. Ge Song, Zide Meng, Fabrice Huet, Frederic Magoules, Lei Yu‡ and Xuelian Lin, " A Hadoop Map Reduce Performance Prediction Method", International Journal of Computer Applications & Information Technology, Vol 2, Issue No 2, Mar 2013.
6. R.Iswarya, P.Saravana Kumar, "NFTaaS on Cloud", International Journal of Latest Trends in Engineering and Technology , Vol 4, Issue 1, May 2014
7. K.Suriya Prakash, P.Saravana Kumar., "Ontology Based Search Engine", International Journal of Latest Trends in Engineering and Technology, Vol 4, Issue 1, May 2014.
8. P.Saravana Kumar, M.Parvathi, M.Kanmani, "Efficient Method for Preventing SQL Injection Attacks on Web Applications Using Encryption and Tokenization", International Journal of Latest Trends in Engineering and Technology, Vol 4, Issue 1, Nov 2014

BIOGRAPHY

Saravana kumar.P is an Assistant Professor in the Computer Application Department, Senthamarai College of Arts and Science, Madurai Kamaraj University, Madurai, India. He received Master of Technology (M.Tech) degree in 2014 from SRM University, Chennai, India, Master of Philosophy in Computer Science (M.Phil) degree in 2011 from PRIST University, Tanjore, India, and Master of Computer Application (MCA) degree in 2010 from SRM University, Chennai, India. His research interests are Data Mining, Web Mining, and Big Data Analytics etc.

Athigopal.M is an Assistant Professor in the Information Technology and Networking Department, Subbalakshmi Lakshmipathy College of Science, Madurai Kamaraj University, Madurai, India. Master of Philosophy in Computer Science (M.Phil) degree in 2012 from PRIST University, Tanjore, India, Bachelor of Education (B.Ed) degree in 2010 from Smt.Savithri College of Education, Trichy, India, Master of Science in Computer Science (M.Sc) degree in 2007 from Madurai Kamaraj University, Madurai, India.

Vetrivel.R.S M.C.A**.,** M.Phil., M.B.A., M.A (H.R.M)., Ph.D is a Vice Principal of Subbalakshmi Lakshmipathy College of Science, Madurai Kamaraj University, Madurai, India.