Bidirectional Data Import to Hive Using SQOOP

MD.Sirajul Huque[1],D.Naveen Reddy[2],G.Uttej[3],K.Rajesh Kumar[4],K.Ranjith Reddy[5]

Assistant Professor, Computer Science & Engineering, Guru Nanak Institution Technical Campus, Hyderabad, India, mailtosirajul@gmail.com [1]

B.Tech Student, Computer Science & Engineering, Guru Nanak Institution Technical Campus, Hyderabad, India, dhondapatinaveenreddy@gmail.com[2]

B.Tech Student, Computer Science & Engineering, Guru Nanak Institution Technical Campus, Hyderabad, India, uttejgumti97@gmail.com[3]

B.Tech Student, Computer Science & Engineering, Guru Nanak Institution Technical Campus, Hyderabad, India, karka.rajeshreddy@gmail.com[4]

B.Tech Student, Computer Science & Engineering, Guru Nanak Institution Technical Campus, Hyderabad, India, kranjit.iiit@gmail.com[5]

Abstract :-Using Hadoop for analytics and data processing requires loading data into clusters and processing it in conjunction with other data that often resides in production databases across the enterprise. Loading bulk data into Hadoop from production systems or accessing it from map reduce applications running on large clusters can be a challenging task. Users must consider details like ensuring consistency of data, the consumption of production system resources, data preparation for provisioning downstream pipeline. Transferring data using scripts is inefficient and time consuming. Directly accessing data residing on external systems from within the map reduce applications complicates applications and exposes the production system to the risk of excessive load originating from cluster nodes. This is where Apache Sqoop fits in. Apache Sqoop is currently undergoing incubation at Apache Software Foundation. More information on this project can be found at http://incubator.apache.org/sqoop. Sqoop allows easy import and export of data from structured data stores such as relational databases, enterprise data warehouses, and NoSQL systems. Using Sqoop, you can provision the data from external system on to HDFS, and populate tables in Hive and HBase.

Keywords: *apache hadoop-1.2.1,apache hive, sqoop, mysql, hive.*

I. INTRODUCTION

Big data exceeds the reach of commonly used hardware environments and software tools to capture, manage, and process it with in a tolerable elapsed time for its user population. It refers to data sets whose size is beyond the ability of typical database software tools to capture, store, manage and analyze. It is a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools.

For the past two decades most business analytics have been created using structured data extracted from operational systems and consolidated into a data warehouse. Big data dramatically increases both the number of data sources and the variety and volume of data that is useful for analysis. A high percentage of this data is often described as multi-structured to distinguish it from the structured operational data used to populate a data warehouse. In most organizations, multi-structured data is growing at a considerably faster rate than structured data. Two important data management trends for processing big data are relational DBMS products optimized for analytical workloads (often called analytic RDBMSs, or ADBMSs) and non-relational systems processing for multi-structured data.

The 3 V's of big data explains the behavior and efficiency of it. In general big data is a problem of unhandled data. Here we propose a new solution to it by using hadoop and its HDFS architecture. Hence here everything is done using hadoop only. These V's are as follows:

1. Volume
2. Velocity
3. Variety

A. Apache Hadoop

software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures

B. Apache Oozie

Apache Oozie is a server based workflow scheduling system to manage Hadoop jobs.

Workflows in Oozie are defined as a collection of control flow and action nodes in a directed acyclic graph. Control flow nodes define the beginning and the end of a workflow (start, end and failure nodes) as well as a mechanism to control the workflow execution path (decision, fork and join nodes). Action nodes are the mechanism by which a workflow triggers the execution of a computation/processing task. Oozie provides support for different types of actions including Hadoop MapReduce, Hadoop distributed file system operations, Pig, SSH, and email. Oozie can also be extended to support additional types of actions.

C. Apache Hive

Apache Hive™ is a data warehouse system for Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in Hadoop-compatible file systems, such as the MapR Data Platform (MDP). Hive provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL. At the same time this language also allows traditional map/reduce programmers to plug in their custom mappers and reducers when it is inconvenient or inefficient to express this logic in HiveQL.

D. Apache Sqoop

Apache Sqoop efficiently transfers bulk data between Apache Hadoop and structured datastores such as relational databases. Sqoop helps offload certain tasks (such as ETL processing) from the EDW to Hadoop for efficient execution at a much lower cost. Sqoop can also be used to extract data from Hadoop and export it into external structured data stores.

II. EXISTING TECHNOLOGY

In the existing technology ,we have the datasets stored .In the HBase database in an unstructured form that have. High latency but less integrity .Here data is more sparse and unclear to analyze. Therefore there is a need to provide a solution to analyze the datasets easily.

III PROPOSED TECHNOLOGY

In the proposed technology, For loading data back to database systems, without any overhead mentioned above. Sqoop works perfect. Sqoop exports the data from distributed file system to database system very optimally. It provides simple command line option, where we can fetch data from different database systems by writing the simple sqoop command.

IV. SQOOP

Hadoop for analytics and data processing requires loading data into clusters.Loading bulk data into Hadoop from production systems or accessing it from map reduce applications running on large clusters can be a challenging task. Transferring data using scripts is inefficient and time consuming. Sqoop allows easy import and export of data from structured data stores such as relational databases, enterprise data warehouses, and NoSQL systems. Using Sqoop, you can provision the data from external system on to HDFS, and populate tables in Hive and HBase. Sqoop integrates with Oozie, allowing you to schedule and automate import and export tasks.

A. Importing Data

The following command is used to import all data from a table called ORDERS from a MySQL database:

```
$ sqoop import --connect jdbc:mysql://localhost/acmedb \
  --table ORDERS --username test --password ****
```

In this command the various options specified are as follows:

- *import:* This is the sub-command that instructs Sqoop to initiate an import.
- *--connect <connect string>, --username <user name>, --password <password>:* These are connection parameters that are used to connect with the database. This is no different from the connection parameters that you use when connecting to the database via a JDBC connection.
- *--table <table name>:* This parameter specifies the table which will be imported.
In the first Step Sqoop introspects the database to gather the necessary metadata for the data being imported. The second step is a map-only Hadoop job that Sqoop submits to the cluster. It is this job that does the actual data transfer using the metadata captured in the previous step.

The imported data is saved in a directory on HDFS based on the table being imported. As is the case with most aspects of Sqoop operation, the user can specify any alternative directory where the files should be populated.

B. Exporting Data

In some cases data processed by Hadoop pipelines may be needed in production systems to help run additional critical business functions. Sqoop can be used to export such data into external datastores as necessary. Continuing our example from above - if data generated by the pipeline on Hadoop corresponded to the ORDERS table in a database somewhere, you could populate it using the following command:

```
$ sqoop export --connect jdbc:mysql://localhost/acmedb \
--table ORDERS --username test --password **** \
--export-dir /user/arvind/ORDERS
```

In this command the various options specified are as follows:

- *export:* This is the sub-command that instructs Sqoop to initiate an export.
- *--connect <connect string>, --username <user name>, --password <password>:* These are connection parameters that are used to connect with the database. This is no different from the connection parameters that you use when connecting to the database via a JDBC connection.
- *--table <table name>:* This parameter specifies the table which will be populated.
- *--export-dir <directory path>:* This is the directory from which data will be exported.

The first step is to introspect the database for metadata, followed by the second step of transferring the data. Sqoop divides the input dataset into splits and then uses individual map tasks to push the splits to the database. Each map task performs this transfer over many transactions in order to ensure optimal throughput and minimal resource utilization.

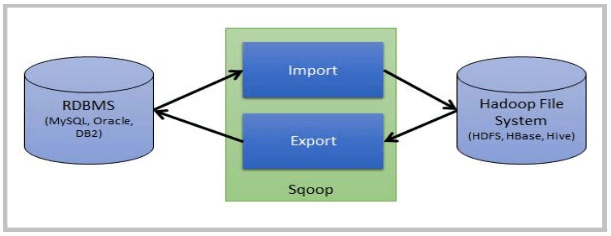


Fig 1: Architecture of SQOOP

C. Working of SQOOP

- **Before you begin**

The JDBC driver should be compatible with the version of the database you are connecting to. Most JDBC drivers are backward compatible, so if you are connecting to databases of different versions you can use the higher version of the JDBC driver. However, for the Oracle JDBC driver, Sqoop does not work unless you are using at least version 11g r2 or later because older version of the Oracle JDBC driver is not supported by Sqoop.

- **About this task**

Sqoop is a set of high-performance open source connectors that can be customized for your specific external connections. Sqoop also offers specific connector modules that are designed for different product types. Large amounts of data can be imported from various relational database sources into a InfoSphere BigInsights cluster by using Sqoop.

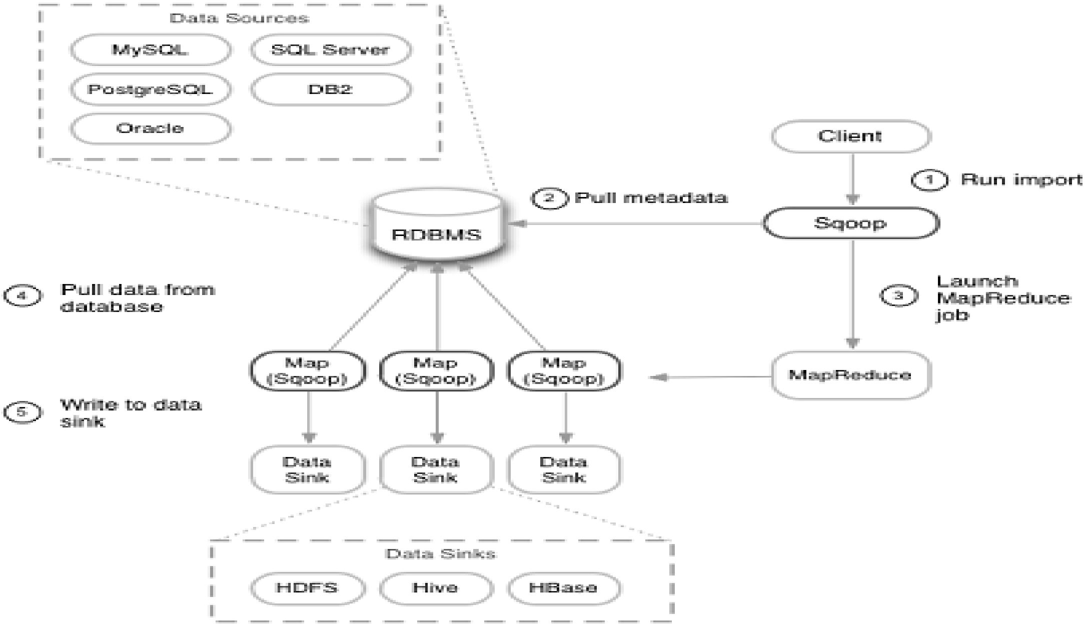


Fig 2: working of sqoop

V. SYSTEM ARCHITECTURE

The systems architect establishes the basic structure of the system, defining the essential core design features and elements that provide the framework for all that follows, and are the hardest to change later. The systems architect provides the architects view of the users' vision for what the system needs to be and do, and the paths along which it must be able to evolve, and strives to maintain the integrity of that vision as it evolves during detailed design and implementation.

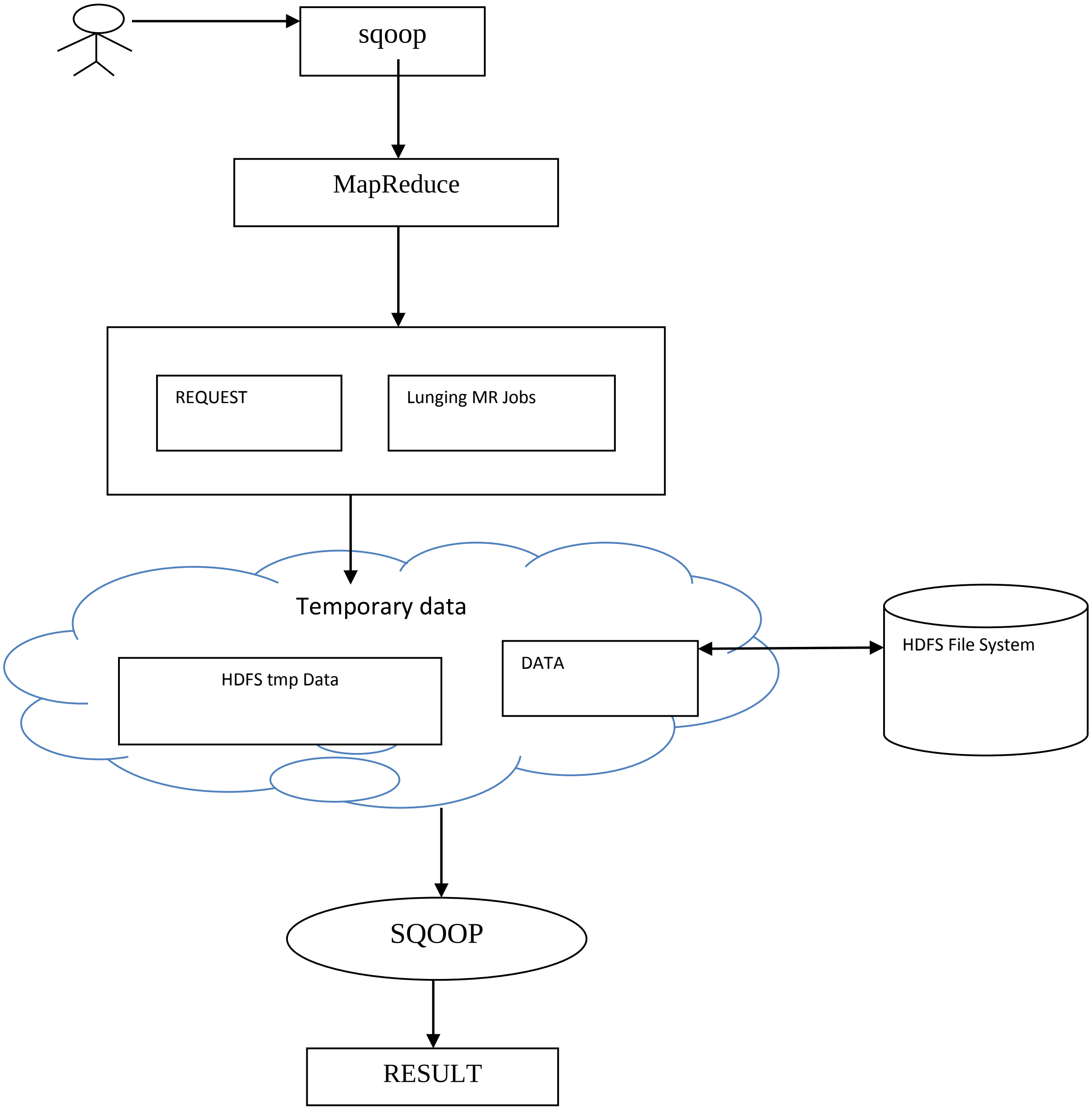
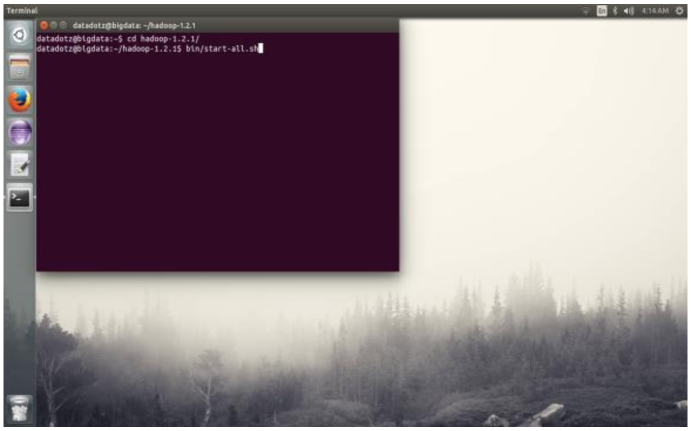


Fig 3: System Architecture

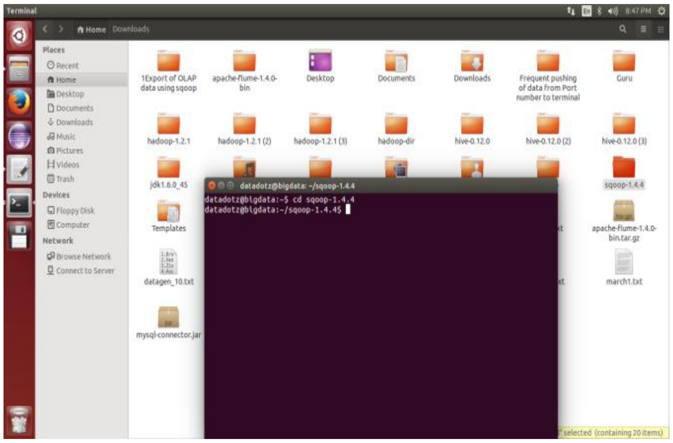
VI. IMPLEMENTATION

1. Open terminal and mount or open hadoop. Then start it by entering bin/start-all.sh



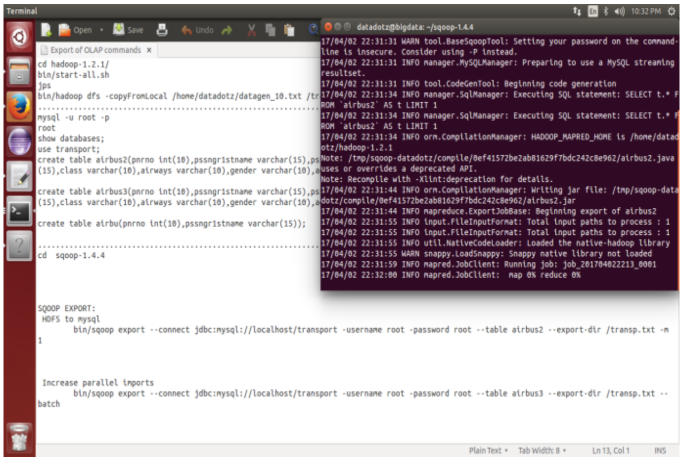
```
datadotz@bigdata:~$ cd hadoop-1.2.1/
datadotz@bigdata:~/hadoop-1.2.1$ bin/start-all.sh
```

2.Open new terminal to run the sqoop – use the command “cd sqoop-1.4.4”



```
datadotz@bigdata:~$ cd sqoop-1.4.4
datadotz@bigdata:~/sqoop-1.4.4$
```

3.Enter the command of Sqoop export – HDFS to MYSQL:



```
cd hadoop-1.2.1/
bin/start-all.sh
jps
bin/hadoop dfs -copyFromLocal /home/datadotz/datagen_10.txt /tr
...............................................................
mysql -u root -p
root
show databases;
use transport;
create table airbus2(pnrno int(10),pssngr1stname varchar(15),ps
(15),class varchar(10),airways varchar(10),gender varchar(10),a

create table airbus3(pnrno int(10),pssngr1stname varchar(15),ps
(15),class varchar(10),airways varchar(10),gender varchar(10),a

create table airbu(pnrno int(10),pssngr1stname varchar(15));

...............................................................
cd  sqoop-1.4.4



SQOOP EXPORT:
 HDFS to mysql
        bin/sqoop export --connect jdbc:mysql://localhost/transport -username root -password root --table airbus2 --export-dir /transp.txt -m
1



 Increase parallel imports
        bin/sqoop export --connect jdbc:mysql://localhost/transport -username root -password root --table airbus3 --export-dir /transp.txt --
batch
```

```
17/04/02 22:31:31 WARN tool.BaseSqoopTool: Setting your password on the command-
line is insecure. Consider using -P instead.
17/04/02 22:31:31 INFO manager.MySQLManager: Preparing to use a MySQL streaming
resultset.
17/04/02 22:31:31 INFO tool.CodeGenTool: Beginning code generation
17/04/02 22:31:34 INFO manager.SqlManager: Executing SQL statement: SELECT t.* F
ROM `airbus2` AS t LIMIT 1
17/04/02 22:31:34 INFO manager.SqlManager: Executing SQL statement: SELECT t.* F
ROM `airbus2` AS t LIMIT 1
17/04/02 22:31:34 INFO orm.CompilationManager: HADOOP_MAPRED_HOME is /home/datad
otz/hadoop-1.2.1
Note: /tmp/sqoop-datadotz/compile/0ef41572be2ab81629f7bdc242c8e962/airbus2.java
uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
17/04/02 22:31:44 INFO orm.CompilationManager: Writing jar file: /tmp/sqoop-data
dotz/compile/0ef41572be2ab81629f7bdc242c8e962/airbus2.jar
17/04/02 22:31:44 INFO mapreduce.ExportJobBase: Beginning export of airbus2
17/04/02 22:31:55 INFO input.FileInputFormat: Total input paths to process : 1
17/04/02 22:31:55 INFO input.FileInputFormat: Total input paths to process : 1
17/04/02 22:31:55 INFO util.NativeCodeLoader: Loaded the native-hadoop library
17/04/02 22:31:55 WARN snappy.LoadSnappy: Snappy native library not loaded
17/04/02 22:31:59 INFO mapred.JobClient: Running job: job_201704022213_0001
17/04/02 22:32:00 INFO mapred.JobClient:  map 0% reduce 0%
```

VII. FUTURE ENHANCEMENTS

Hadoop is uniquely capable of storing, aggregating, and refining multi-structured data sources into formats that fuel new business insights. Apache Hadoop is fast becoming the defector platform for processing Big Data.

Hadoop started from a relatively humble beginning as a point solution for small search systems. Its growth into an important technology to the broader enterprise community dates back to Yahoo’s 2006 decision to evolve Hadoop into a system for solving its internet scale big data problems. Eric will discuss the current state of Hadoop and what is coming from a development standpoint as Hadoop evolves to meet more workloads.

Improvements and other components based on Sqoop Unit that could be built later.

For example, we could build a SqoopTestCase and Sqoop TestSuite on top of SqoopTest to:

1. Add the notion of workspaces for each test.

2. Remove the boiler plate code appearing when there is more than one test methods.

VIII. CONCLUSION

While this tutorial focused on one set of data in one Hadoop cluster, you can configure the Hadoop cluster to be one unit of a larger system, connecting it to other systems of Hadoop clusters or even connecting it to the cloud for broader interactions and greater data access. With this big data infrastructure, and its data analytics application, the learning rate of the implemented algorithms will increase with more access to data at rest or to streaming data, which will allow you to make better, quicker, and more accurate business decisions.

With the rise of Apache Hadoop, next-generation enterprise data architecture is emerging that connects the systems powering business transactions and business intelligence. Hadoop is uniquely capable of storing, aggregating, and refining multi-structured data sources into formats that fuel new business insights. Apache Hadoop is fast becoming the defector platform for processing Big Data.

REFERENCES

WEB RESOURCES

1) https://hadoop.apache.org/docs/r1.2.1/single_node_setup.html
2) http://www.tutorialspoint.com/hbase/hbase_overview.htm
3) https://cwiki.apache.org/confluence/display/Hive/Getting Started
4) http://hbase.apache.org/0.94/book.html#getting_started.
5) http://www.tutorialspoint.com/hadoop/hadoop_introduction.htm
6) http://www.tutorialspoint.com/hive/hive_introduction.htm

BIBILOGRAPHY

1. Mukherjee, A.; Datta, J.; Jorapur, R.; Singhvi, R.; Haloi, S.; Akram, W., (18-22 Dec.,2012) , “Shared disk big data analytics with Apache Hadoop”
2. Aditya B. Patel, Manashvi Birla, Ushma Nair ,(6-8 Dec. 2012),“Addressing Big Data Problem Using Hadoop and Map Reduce”
3. Yu Li ; WenmingQiu ; Awada, U. ; Keqiu Li,,(Dec 2012),” Big Data Processing in Cloud Computing Environments”
4. Garlasu, D.; Sandulescu, V. ; Halcu, I. ; Neculoiu, G. ;,(17-19 Jan. 2013),”A Big Data implementation based on Grid Computing”, Grid Computing
5. Sagiroglu, S.; Sinanc, D. ,(20-24 May 2013),”Big Data: A Review”
6. Grosso, P. ; de Laat, C. ; Membrey, P.,(20-24 May 2013),” Addressing big data issues in Scientific Data Infrastructure”
7. Kogge, P.M.,(20-24 May,2013), “Big data, deep data, and the effect of system architectures on performance”
8. Szczuka, Marcin,(24-28 June,2013),” How deep data becomes big data”
9. Zhu, X. ; Wu, G. ; Ding, W.,(26 June,2013),” Data Mining with Big Data”
10. Zhang, Du,(16-18 July,2013),” Inconsistencies in big data”
11. Tien, J.M.(17-19 July,2013),” Big Data: Unleashing information”
12. Katal, A Wazid, M. ; Goudar, R.H., (Aug,2013),” Big data: Issues, challenges, tools and Good practices”