Chapter 7

Using Hadoop Cluster and MapReduce
7. Using Hadoop Cluster and MapReduce for Big Data Problems

The size of the databases used in today’s enterprises has been growing at exponential rates day by day. In this electronic age, increasing number of organizations are facing the problem of explosion of data and the size of the databases used in today’s enterprises has been growing at exponential rates. Data is generated through many sources like business processes, transactions, social networking sites, web servers, etc. and remains in structured as well as unstructured form [176].

Today's business and scientific applications are having enterprise features like large scale, data-intensive, web-oriented and accessed from diverse devices including mobile devices. Simultaneously, the need to process and analyze the large volumes of data for business decision making has also increased. Processing or analyzing the huge amount of data or extracting meaningful information is a challenging task. In several business and scientific applications, there is a need to process terabytes of data in efficient manner on daily bases. This has contributed to the big data problem faced by the industry due to the inability of conventional database systems and software tools to manage or process the big data sets within tolerable time limits.

Big Data and Related problems

The term “Big Data” refers to large sized data sets which are difficulty to storage, process and analyze by currently available tools and database management systems, within reasonable time limit constraints. Big data sizes in today’s organizations are constantly increasing - from terabytes to petabytes of data [177].

Big data also includes the unstructured and semi-structured data like web logs, RFID generated data, sensor networks, video surveillance data in the organizations, audio, video and graphical data, satellite and geo-spatial data, social data from social networks, Internet data and web pages, Internet search indexing, domain specific big data like biological life science data, banking and financial portfolios, telephone call records,
national security surveillance, medical images and records etc. “Big Data scenarios are faced by many organizations like Walmart (which handles billions of transaction per day resulting into database size of multiple petabytes), facebook and twitter like sites which have to handle billions of unstructured data like photos, videos, text feeds and messages. In addition to variations in the amount of data stored in different sectors, the types of data generated and stored—i.e., whether the data encodes video, images, audio, or text/numeric information—also differ markedly from industry to industry [178].”

The big data problem emerges due to the inability of currently available database management systems and software tools to store and process large amounts of structured and unstructured data. Processing of data can include various operations depending on usage like culling, tagging, highlighting, indexing, searching, faceting, etc operations. It is not possible for single or few machines to store or process this huge amount of data in a finite time period.

To solve the above problems, I have done prototype implementation of Hadoop cluster in grid test bed, HDFS storage and Map Reduce framework for processing large data sets by considering compute and big data application scenario. The results obtained from various experiments indicate favorable results of above approach to address the big data problem.
Chapter 7: Using Hadoop Cluster and MapReduce Framework in the Grid

7.1 Hadoop and Map Reduce Framework

Hadoop is a software framework developed by apache open source foundation for creating cluster and parallel processing of data intensive applications in a for reliable, scalable and fault tolerant manner. The framework includes data cluster, distributed data storage and parallel processing model called Map Reduce. Hadoop is highly scalable to work with thousand of processing nodes and petabytes of data.

HDFS (Hadoop Distributed File System)

“The Hadoop Distributed File System (HDFS) runs on commodity hardware and provide high performance distributed file system which is reliable and fault tolerant. HDFS is designed for big data applications that needs to process large amount of heterogeneous data using high throughput concurrent and parallel access and processing. HDFS stores the data sets in parallel using large number of servers/commodity machines, which makes it possible to perform parallel processing of local data available in nodes (Map reduce jobs). HDFS has master/slave architecture. At the loading time, big sized data is split into parts which are stored and processed by different nodes in the hadoop cluster [179].”

MapReduce Framework

MapReduce is a parallel programming framework based on the model published by Google in 2004 to support distributed computing on large data sets on compute clusters. It provides parallel programming model for parallel and distributed processing of large data sets.

“Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs and a reduce function that merges all intermediate values associated with the same intermediate key [180].”
The problem needs to be implemented in Map Reduce framework steps which operates on list based processing of data. The working of Map Reduce steps is as under [181] [182]:

"Map" step:

Master node filters, transforms the input and assigns the partitions of input data to worker nodes. Mapping step processes the input data and generates the intermediate result. The mapping step is done in parallel manner by multiple slave nodes.

\[ \text{Map} \ (k_1, v_1) \rightarrow \text{list} \ (k_2, v_2) \]
"Reduce" step:

The intermediate results from multiple mapping processes are combined by reduce process. Reduce function is also applied in parallel manner to each group of intermediate results with same key values.

Reduce \((K2, \text{list}(v2)) \rightarrow \text{list}(v3)\)

**mapper** (filename, file-contents):

- **for each** word in file-contents:
  - **emit** (word, 1)

**reducer** (word, values):

- \(\text{sum} = 0\)
- **for each** value in values:
  - \(\text{sum} = \text{sum} + \text{value}\)
- **emit** (word, sum)

Figure 7.2: Pseudo code of Map Reduce for counting words in different files program [183]
Chapter 7: Using Hadoop Cluster and MapReduce Framework in the Grid

7.2 Experimental Setup and System Architecture

The system architecture comprises of multiple node cluster having one master node, multiple slave nodes, HDFS name node and multiple HDFS data node for storing data. The HDFS architecture consists of single name node running on master node which manages the meta-data and access to the file blocks stored by multiple data nodes running on slave nodes. The data is divided into file blocks and stored in distributed manner by different data nodes. The file blocks are automatically replicated (default replication factor is 3) and redundantly by different data nodes to ensure high available and fault tolerance.

![Hadoop Cluster Setup using Master and Slave nodes](image)

Figure 7.3: Hadoop Cluster Setup using Master and Slave nodes

For performing the big data experiments, setup of Hadoop data cluster comprising of four nodes and Hadoop Distributed File System (HDFS) for storage was used. Before moving to multi-node cluster, single node cluster was first configured and tested.
Hadoop has many configuration parameters, but the most relevant for the purpose of this evaluation is the number of concurrent Map and Reduce tasks that are allowed to run on each node. We configured our cluster to run eight concurrent tasks per worker node. Each Map/Reduce program that is run is partitioned into M map tasks and R reduce tasks. Input and output data for the Map/Reduce programs is stored in HDFS, while input and output data for the data-parallel stack-based implementation is stored directly on the local disks.

One node was configured as Master node and other nodes were designated as slave nodes. The master node runs the “master” daemons: NameNode for the HDFS storage layer and JobTracker for the MapReduce processing layer. The slave nodes run the “slave” daemons: DataNode for the HDFS layer and TaskTracker for MapReduce processing layer. The master node was also used as slave node to increase the processing nodes. The software used for master and slave nodes was Sun Java 1.6, Ubuntu Linux 10.04 LTS and hadoop 1.0.3.
Chapter 7: Using Hadoop Cluster and MapReduce Framework in the Grid

7.3 Experiment Results

7.3.1. Text processing application

The first experiment was a simple text processing word count experiment to count the number of words that occur within a set of large sized documents. Different documents were loaded on different nodes. The map function runs on each node and processes the document stored locally and generates intermediate result in form of (“word”, 1). The shuffle step organizes and sorts all the intermediate results generated by the Map process based on same key value, and all results with same key value are passed on to same reducer process. The reducing process, accepts the intermediate results with same key (word), apply the aggregation logic i.e. count and add the number of words occurred in all the documents.

1) Experiment with increase in number of nodes

Dataset: Size of files used = 100 Mb

![Figure 7.4: Execution time with varying number of nodes](image)

2) Experiment with increase in size of dataset and nodes

Modeling and Prototyping of RMS for QoS Oriented Grid
The above experiment results indicate that there is significant decrease in execution time with increase in number of nodes used in Hadoop cluster.

Figure 7.5: Execution time with varying dataset and nodes
7.3.2. Earthquake Data Analysis

In the second experiment, we have analyzed the earthquake data published by U.S. Geological Survey (USGS). The earthquake data is available in the form of CSV (comma-separated values) files published at periodic intervals. The analysis of earthquake data provides an answer to where the earthquakes occurred by location, number of earthquakes on particular date.

1) Experiment with increase in number of nodes

Dataset - 7 day Earthquake Report of USGS

![Graph showing the relationship between the number of nodes and execution time.](image)

Figure 7.6: Quake analysis – No. of nodes v/s Execution time
2) Experiment with increase in size of dataset and nodes

![Graph showing Total Time in Seconds against Number of Days Earthquake Report Published for 2 nodes and 4 nodes.]

Figure 7.7: Earth Quake analysis – No. of days report v/s Execution time

The above experiment results indicates that Hadoop and Map reduce setup is highly scalable to process large datasets and job execution time is decreased with the use of more number of nodes in the hadoop cluster.

In this experimental work, Hadoop cluster, HDFS and Map Reduce programming framework is used to address data intensive application scenarios. The results obtained from various experiments indicate favorable results of above approach to address big data problem. The above experiment result indicates that Hadoop cluster is scalable to support increased dataset and takes less job execution time with increase in number of nodes.