Original Article

Hadoop MapReduce: A Programming Model for Large Scale Data Processing

Sunita B Aher* and Anita R. Kulkarni

Department of Computer Science & Engineering, Walchand Institute of Technology, Solapur, India

Address for Correspondence

Department of Computer Science & Engineering, Walchand Institute of Technology, Solapur, India.
E-mail: sunita_aher@yahoo.com

ABSTRACT

Hadoop is free open source framework for Cloud Computing Environment. It is used to implement Google[tm] MapReduce framework. Map-Reduce technique is a popular framework which is used to process and generate large data on cloud. Map-Reduce technique of Hadoop is used for large-scale data-intensive applications like data mining and web indexing. If the problem is modelled as MapReduce problem then it is possible to take advantage of computing environment provided by Hadoop.
A Map-Reduce job usually splits the input data-set into independent unit. These units are processed in parallel manner and combine the result to obtain the final result. This paper presents the technique of Map-Reduce framework of Hadoop. We also present the steps to execute the program on Hadoop and explained result that we obtained using MapReduce technique of Hadoop.

Keywords: Cloud computing, Hadoop, MapReduce framework, Sort and Shuffle, Key-value pair.

INTRODUCTION

Now a day, data and web data are increasing in terms of 10 to 100 terabytes which cannot be mine or process on single server. Yahoo and Apache developed Hadoop which replicates the same data 3 times and distributes the pieces to several systems connected in the network. So, if one system goes down other 2 replicas are available. So, it's cheap, robust and fault tolerant. Cloud BLAST, a distributed implementation of NCBI BLAST using Hadoop[1] is investigated an efficient approach to the execution of bioinformatics applications.

The 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[12]. The project includes modules like Hadoop Common, Hadoop Distributed File System (HDFS™), Hadoop YARN and Hadoop Map-Reduce. Other Hadoop-related projects at Apache include Ambari™, Avro™, Cassandra™, Chukwa™ etc.

How MapReduce technique of Hadoop are used for scientific data analysis and bioinformatics are explained in[2,3]. Cloud Burst uses the open-source Hadoop implementation of MapReduce to parallelize execution using multiple compute nodes[5]. The research in[7] adds to Map-Reduce Merge phase that can efficiently merge data already partitioned and sorted by map and reduce modules. Kyong-Ha Lee[8] characterizes the MapReduce framework and discuss its inherent pros and cons. The paper[9] discusses the opportunities and challenges for efficient parallel data processing in clouds and present research project Nephele. Dominic Battré *et al.*[10] designed the PACT programming model which is a generalization of the well-known map/ reduce programming model.

This paper presents the technique of Map-Reduce framework of Hadoop. We also present the steps to execute the program on Hadoop and explained result that we obtained using MapReduce technique of Hadoop.

Remaining part of the paper is arranged as follows. Section 2 discusses about Hadoop and the MapReduce framework of Hadoop is presented in section 3. Section 4 discuss about the running the application on Hadoop followed by the result in section 5. Section 6 presents the Hadoop on the cloud. Section 7 discusses the conclusion followed by the references.

Hadoop

Apache Hadoop is an open-source software framework for distributed computing which is used for storage and large-scale processing of data-sets on clusters of commodity hardware.

The Apache Hadoop framework is composed of the following modules:

- Hadoop Common – It consist of libraries and utilities needed by other Hadoop modules.
- Hadoop Distributed File System (HDFS) – It is a distributed file-system that stores data on commodity machines that gives very high aggregate bandwidth across the cluster. It provides high-throughput access to application data.
- Hadoop YARN – This module is a resource-management platform. This module is responsible for managing compute resources in clusters and using them for scheduling of users' applications.
- Hadoop MapReduce – This is a programming model for large scale data processing.

Apache Hadoop's MapReduce and HDFS components are derived from Google's MapReduce and Google File System (GFS) respectively.

A small Hadoop cluster includes a single master and multiple worker nodes. The master node consists of a JobTracker, TaskTracker, NameNode and DataNode. A worker node acts as both a DataNode and TaskTracker. In a larger cluster, the HDFS is managed through a dedicated NameNode server to host the file system index, and a secondary NameNode that can generate snapshots of the namenode's memory structures and prevent file-system corruption and reducing loss of data. JobTracker server can manage job scheduling[11].

Hadoop implements a Map/Reduce, where the application is divided into many small unit of work, each of which are executed or re-executed on any node in the cluster. It provides a distributed file system (HDFS) that stores data on the compute nodes which provides very high bandwidth across the cluster. Both MapReduce and the Hadoop Distributed File System are designed so that node failures if occurs, are automatically handled by the framework[13].

The multi node Hadoop cluster is shown in figure 1.

Hadoop-MapReduce framework

Hadoop Map-Reduce is a software framework used to write applications which process vast amounts of data in-parallel on large clusters of commodity hardware in a reliable, fault-tolerant manner.

A Map-Reduce job usually splits the input data-set into independent unit or chunks. These chunks are processed in isolation by tasks called Mappers. The outputs from the mappers denoted as intermediate outputs (IOs) are given as input to the second set of tasks called Reducers. The process of bringing together IOs into a set of Reducers is known as shuffling process. The Reducers produce the final outputs (FOs)[6].

Overall a MapReduce program consists of two phases:

The map phase

a. The master node takes the input.
b. It divides the input into smaller sub-problems.
c. The master node distributes these smaller sub-problems to worker nodes.
d. A worker node may do this again to lead to a multi-level tree structure.
e. The worker node processes the smaller problem.
f. It passes the answer back to its master node.

The reduce phase

a. The master node collects the answers to all the sub-problems given by worker nodes.
b. It combines all answers to form the output to the original problem.

Detailed steps to be followed in MapReduce technique are as follows:

1. **Preparing the Map input** – The system selects the Map processors, allocates them the input key value K1 to work on, and provides that processor with all the input data associated with that key value.
2. **Execute the user-provided Map () code** – Map () code is executed exactly once for each K1 key value, generating output that is organized by key values K2.
3. **"Shuffle" the output of the Map() to the Reduce processors** – the Map-Reduce system selects the Reduce processors, assigns the K2 key value to work on and provides that processor with all the data generated by Map() associated with that key value.
4. **Execute the Reduce () code provided by the user** – Reduce () is run exactly once for each K2 key value produced by the Map step.
5. **Produce the final output** – the MapReduce system collects all the output generated by, Reduce () and sorts it by key value K2 to produce the final outcome[6].

Figure 2 shows the MapReduce phases.

Consider the example of subjects for Computer Science and Engineering course: ACA (Advanced Computer Architecture), ADS (Advanced Database System), CC (Compiler Construction), DBMS (Database Management System), DMS (Discrete Mathematical Structure), NS (Network Security), TOC (Theory of Computation).

Two datasets with different combinations of subjects are

Dataset 1: TOC DMS CC
CC DBMS ADS
DMS ACA NS

Dataset 2: TOC DBMS CC
TOC ADS DMS
ADS ACA TOC

Map step

For each of the record in dataset, map (String key, String value) that is map (k1,v1) → list(k2,v2) is produced as follows.

TOC DMS CC {("TOC","1"), ("DMS","1"), ("CC","1")}

CC DBMS ADS {("CC","1"), ("DBMS","1"), ("ADS","1")}
DMS ACA NS {("DMS","1"), ("ACA","1"), ("NS","1")}
TOC DBMS CC {("TOC","1"), ("DBMS","1"), ("CC","1")}
TOC ADS DMS {("TOC","1"), ("ADS","1"), ("DMS","1")}
ADS ACA TOC {("ADS","1"), ("ACA","1"), ("TOC","1")}

Reduce step

For each of the above Map result, the obtained reduce (String key, Iterator values) that is reduce (k2, list (v2)) → list (v2) is as follows:

reduce ("ACA",<1, 1>) → 2
reduce ("ADS",<1, 1, 1>) → 3
reduce ("CC",<1, 1, 1, 1>) → 4
reduce ("DBMS",<1, 1>) → 2
reduce ("DMS",<1, 1, 1>) → 3
reduce ("NS",<1>) → 1
reduce ("TOC",<1, 1, 1, 1>) → 4

So phases of MapReduce technique with example is shown in figure 3.

Running data sets on Hadoop

Running the dataset on Hadoop is given as[6]:

1. Put your input data files in the folder named "input"; in the instructions below the directory is assumed to be in your home directory.
2. Output folder is named "output". Do not explicitly create the folder, it may give an error. Just give it's location in the command as below. For subsequent runs, you may need to delete the folder before running the program below.
3. Go to the hadoop directory (cd /users/fac/sudarsha/Hadoop). Run the program using: ./bin/hadoop jar ~/WordCount.jar ~/input ~/output (assuming Jar file, input folder and output folder are in your home directory).
4. Output is created in the file part-00000 in the output folder.

RESULTS

Here we consider two files check and name[4] given in first row of table 1. Row two of table 1 shows the result of running these dataset on Hadoop using MapReduce technique.

Hadoop on cloud

Cloud computing refers to the delivery of computing resources over the Internet. Examples of cloud services include online file storage, social networking sites, webmail, and online business applications and many more[16]. Cloud computing is used various field like Entertainment, Security issue, Military Operations, Business, finance, Medical etc. Following are reasons for why Hadoop in the cloud makes sense[14]:

- **Low Cost of Innovation:** Running Hadoop on the cloud has same advantage as running any other software offering on the cloud. The cloud also makes sense for a quick, one time use case involving big data computation.
- **Large scale resources are obtained quickly:** Google knew that there would be need for more and more hardware resources for processing huge amount of data. As the analytics demand within enterprises grew, there was a need to expand the capacity of the Hadoop clusters. The cloud, with instant access to hardware resources, is the solution to provide a platform that scales fast to meet growing needs of business.
- **Batch workloads are handled efficiently:** With a batch-oriented system, Hadoop involve processing scheduled jobs for new incoming data on a fixed, temporal basis. Companies collect data from devices or web server and input this data into analytics application on Hadoop.

The cloud is more efficient to handle such batch workloads.

- **Variable resource requirements are handled:** The requirement of all Hadoop jobs is not the same. Some of them require some a lot of I/O bandwidth, while some require more compute resources and some require more memory. A physical Hadoop cluster is built of homogeneous machines which handle the largest job.

Cloud solutions offer a choice to the end user to provide clusters with different types of machines for different types of workloads.

For example, Amazon Elastic MapReduce can be used to launch a cluster with m2.large machines if Hadoop jobs require more memory and c1.xlarge machines if Hadoop jobs require intensive computations.

- **Running closer to the Data:** As businesses move their services to the cloud, data starts living on the cloud. As analytics thrives on large amount of data, running Hadoop clusters in the cloud environment is an efficient solution to this problem.
- **Hadoop operations are simplified:** As cluster consolidation happens in the enterprise, all user jobs get bunched up in a shared cluster. Hence the administrators of the cluster face the problem such as user jobs interfering with one another, varied security constraints etc.

The cloud can provide different types of clusters with different characteristics and configurations, suitable for a particular set of jobs.

Following are options for running Hadoop in the Cloud[15]:

- **Hadoop as a Service in the Public Cloud** – Hadoop distributions such as Cloudera CDH, MapReduce, IBM BigInsights, Hortonworks can be launched and run on the public clouds like AWS, Rackspace, MS Azure, IBM SmartCloud, etc. that offer Infrastructure as a Service (IAAS).
- **MapReduce as a Service** – Amazon's EMR such as Elastic MapReduce provides a way to run MapReduce jobs without having to install a Hadoop cluster on its cloud.
- **Hadoop on S3** – You can run Hadoop using Amazon's S3 to store data. Netflix uses a Hadoop cluster using S3.
- **Hadoop in private Cloud** –in case of a private cloud, there is more control over infrastructure that will allow provisioning bare-metal servers or creating a separate isolated network for your Hadoop clusters. Private cloud solutions support Paas service that offers patterns for deploying Hadoop clusters easily. IBM offers patterns for deploying InfoSphere Big Insights on their SmartCloud Enterprise.

CONCLUSION

In this paper, we review the Map-Reduce technique of Hadoop which is the free open source software for the cloud computing environment. We also presents the steps to execute the dataset consisting of different combination of subjects of Computer Science and Engineering course on Hadoop and explained result that we obtained using MapReduce technique of Hadoop. Cloud computing involves a large number of computers connected through Internet. So the reasons for why Hadoop in the cloud makes sense and options for running Hadoop in the Cloud are discussed.

REFERENCES

1. A. Matsunaga *et al*., Cloudblast: Combining mapreduce and virtualization on distributed resources for bioinformatics applications, in Fourth IEEE International Conference on eScience, pages 222–229, 2008.

2. J. Ekanayake *et al*., Mapreduce for data intensive scientific analyses. In the 4th IEEE International Conference on eScience, pages 277–284, 2008.
3. R. Taylor., An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics, BMC bioinformatics, 2010.
4. http://www.it.iitb.ac.in/moodle/course/view.php?id=74.
5. M.C. Schatz. CloudBurst: highly sensitive read mapping with MapReduce. Bioinformatics, 25(11):1363, 2009.
6. http://en.wikipedia.org/wiki/MapReduce
7. H. Yang *et al*., Map-reduce-merge: simplified relational data processing on large clusters, In Proceedings of the 2007 ACM SIGMOD, pages 1029–1040, 2007.
8. Kyong-Ha Lee *et al*., Parallel Data Processing with MapReduce: A Survey, SIGMOD Record, December 2011 (Vol. 40, No. 4).
9. D. Warneke *et al* . Nephele: efficient parallel data processing in the cloud. In Proceedings of the 2nd MTAGS, pages 1–10, 2009.
10. D. Battr´e *et al* . Nephele/PACTs: a programming model and execution framework for web-scale analytical processing. In Proceedings of the 1st ACM symposium on Cloud computing, pages 119–130, 2010.
11. http://en.wikipedia.org/wiki/Apache_Hadoop
12. http://hadoop.apache.org/
13. http://wiki.apache.org/hadoop/
14. http://www.thoughtworks.com/insights/blog/6-reasons-why-hadoop-cloud-makes-sense
15. http://www.ibmbigdatahub.com/blog/running-hadoop-cloud.
16. Introduction to Cloud Computing, accessed on 11-03-2014 from http://www.priv.gc.ca/resource/fs-fi/02_05_d_51_cc_e.pdf.

Table 1. Files used and result using MapReduce technique of Hadoop

File Check In computer science, an inverted index (also referred to as postings file or inverted file) is an index data structure storing a mapping from content, such as words or numbers, to its locations in a database file, or in a document or a set of documents. The purpose of an inverted index is to allow fast full text searches, at a cost of increased processing when a document is added to the database. The inverted file may be the database file itself, rather than its index. test:output&hello@echo*iitb'asd?why File Name rajesh satish rajesh satish ledu ledu rajesh database is the tough	
In	{check 1}
The	{check 47} {check 74}
a	{check 61} {check 22} {check 40} {check 67} {check 43} {check 35}
added	{check 70}
allow	{check 55}
also	{check 7}
an	{check 4} {check 50} {check 17}

as {check 10} {check 27}
asd {check 92}
at {check 60}
be {check 78}
computer {check 2}
content {check 25}
cost {check 62}
data {check 19}
database {check 73} {check 80} {check 36} {names 1}
document {check 68} {check 41}
documents {check 46}
echo {check 90}
fast {check 56}
file {check 15} {check 76} {check 12} {check 81} {check 37}
from {check 24}
full {check 57}
hello {check 89}
iitb {check 91}
in {check 39} {check 34}
increased {check 64}
index {check 6} {check 18} {check 86} {check 52}
inverted {check 5} {check 51} {check 75} {check 14}
is {check 53} {check 69} {check 16} {names 2}
its {check 32} {check 85}
itself {check 82}
ledu {names 1} {names 1}
locations {check 33}
mapping {check 23}
may {check 77}
numbers {check 30}
of {check 49} {check 45} {check 63}
or {check 42} {check 13} {check 29} {check 38}
output {check 88}
postings {check 11}
processing {check 65}
purpose {check 48}
rajesh {names 1} {names 1} {names 1}
rather {check 83}
referred {check 8}
satish {names 1} {names 1}
science {check 3}
searches {check 59}
set {check 44}
storing {check 21}
structure {check 20}
such {check 26}
test {check 87}
text {check 58}

than	{check 84}
the	{check 72} {check 79} {names 3}
to	{check 9} {check 31} {check 71} {check 54}
tough	{names 4}
when	{check 66}
why	{check 93}
words	{check 28}

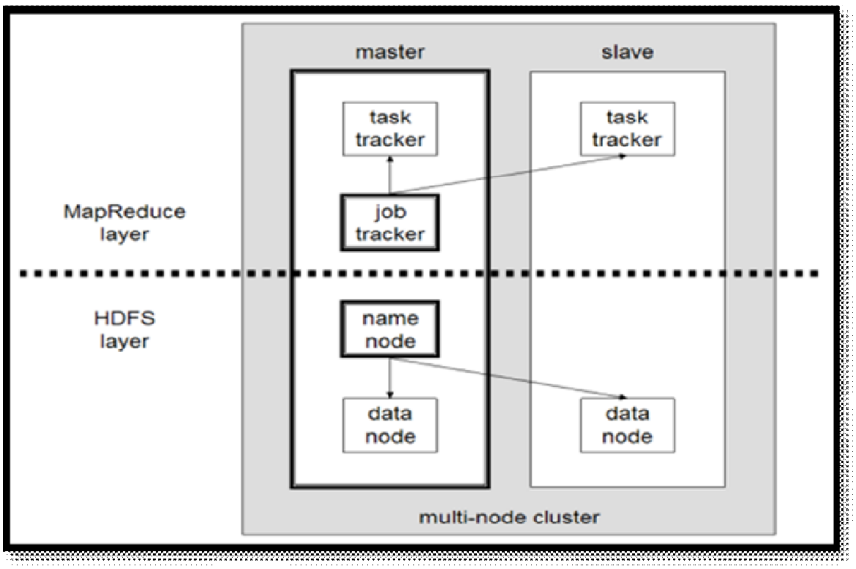

Figure 1. Multi-node Hadoop cluster[11]

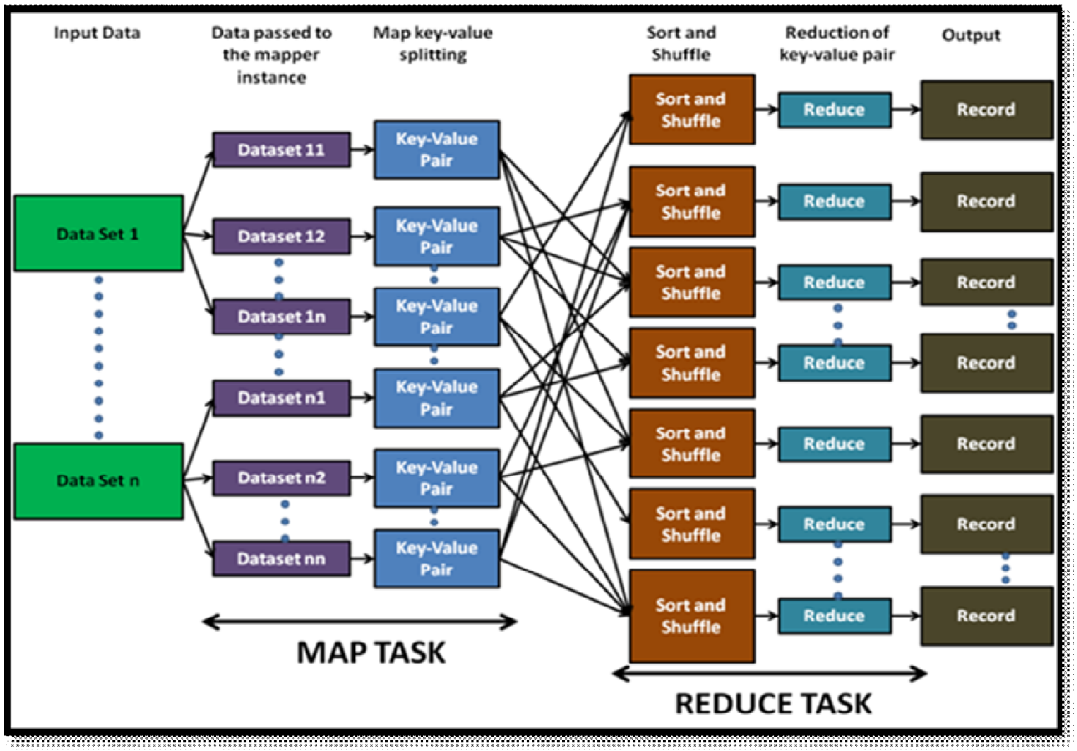

Figure 2. MapReduce Phases

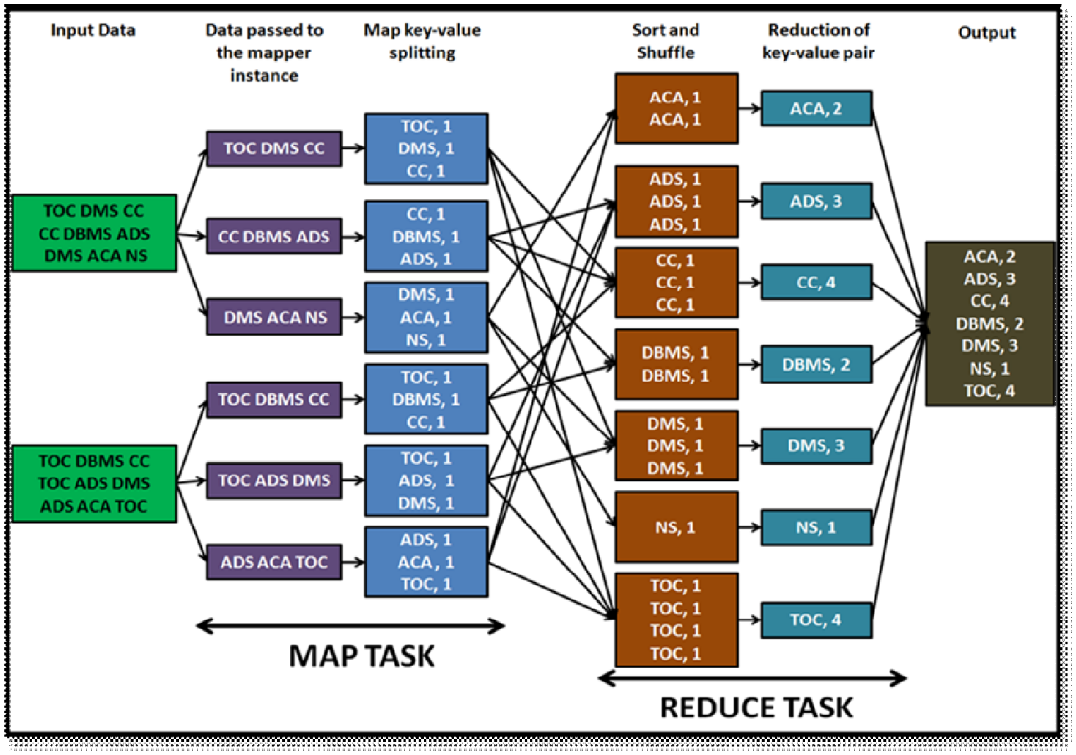

Figure 3. Working of MapReduce phases with example