AN ANALYSIS OF E HADOOP/MAPREDUCE/H BASE FRAMEWORK AND ITS CURRENT APPLICATIONS IN BIOINFORMATICS

Ramesh Chandra Tripathi*

*Professor,
Department of Computer Engineering, Teerthanker Mahaveer University,
Moradabad, Uttar Pradesh, INDIA
Email id: tripathi.computers@tmu.ac.in

DOI: 10.5958/2249-7137.2021.02501.5

ABSTRACT

High-performance computing (HPC) has become more essential in bioinformatics data processing as a result of new computational difficulties. Work is usually distributed over a cluster of computers that connect to a shared file system housed on a storage area network. The Message Passing Interface (MPI) and, more recently, Hadoop's MapReduce API have been used to achieve work parallelization. Cloud computing is another computer architecture/service model that is currently being investigated. In a nutshell, cloud computing is HPC with a web interface plus the flexibility to scale up and down quickly for on-demand usage. Remote clients upload potentially large data sets for analysis in the Hadoop framework or other parallelized environments running in the data center, with the server side deployed in data centers working on clusters. The present use of Hadoop, a toplevel Apache Software Foundation project, and related open source software projects in the bioinformatics field is discussed. The principles underlying Hadoop and the HBase project are explained, as well as the existing bioinformatics software that uses Hadoop. The emphasis is on next-generation sequencing, which is now the most popular application area.

KEYWORDS: *API, Hadoop, H Base, Map Reduce, Pig.*

1. INTRODUCTION

1.1 Hadoop:

Hadoop is a software framework for large-scale distributed data processing that may be deployed on a commodity Linux cluster. Other than potential adjustments to satisfy minimum suggested RAM, disk space, and other requirements per node, no hardware modifications are required. Doug Cutting (named after his son's pet elephant) developed the first version of Hadoop in 2004. In January 2008, Hadoop was designated as a top-level Apache Software Foundation project. There have been numerous academic and commercial contributions (Yahoo being the biggest), and Hadoop has a huge and fast expanding user community **[1].**

a) Components:

Hadoop includes a Java-based API that enables parallel processing across cluster nodes using the MapReduce paradigm, as well as the robust, fault-tolerant Hadoop Distributed File System

(HDFS), which was influenced by Google's file system. Hadoop Streaming, a tool that enables users to build and execute jobs using any executable as the mapper and/or reducer, lets users to utilize code written in other languages, such as Python and C. Hadoop also has Job and Task Trackers, which monitor the execution of applications throughout the cluster's nodes [**2]**.

b) Data locality:

Hadoop attempts to synchronize the data with the compute node automatically. Hadoop schedules Map jobs near to the data they'll be working with, with "close" implying the same node or, at the very least, the same rack. Hadoop's performance is heavily influenced by this. A Hadoop application operating on a 910-node cluster set a world record in April 2008, sorting a terabyte of data in under 3.5 minutes. As Hadoop has evolved, it has continued to increase its speed.

c) Map Reduce paradigm:

Hadoop implements its fault-tolerant distributed computing system across huge data sets stored in the cluster's distributed file system using a Map/ Reduce execution engine. This MapReduce technique was pioneered by Google, was recently patented for usage on clusters by Google and licensed to Apache, and is currently being developed by a large group of researchers. There are distinct Map and Reduce stages, each of which operates on sets of key-value pairs and is performed in parallel. As a result, program execution is split into two stages: Map and Reduce, which are separated by data transfers between cluster nodes. A node performs a Map function on a portion of the input data in the first stage. The map output is a collection of records stored on that node in the form of keyvalue pairs. The records for each particular key, which may be distributed over many nodes, are gathered at the node that runs the Reducer for that key. This entails machine-to-machine data transmission. This second Reduce step can't start until all of the data from the Map stage has been delivered to the correct computer. As a final result, the Reduce step generates a new collection of key-value pairs. This is a basic programming paradigm that just uses key-value pairs, yet it can accommodate a surprising amount of jobs and algorithms [**3]**.

d) HDFS file system:

There are several disadvantages to using HDFS. HDFS isn't as good as a conventional relational database management system at handling continuous changes (write many). Furthermore, HDFS cannot be mounted directly on a current operating system. As a result, transferring data into and out of the HDFS file system may be difficult. There are many open source projects developed on top of Hadoop, in addition to Hadoop itself **[4].**

1.2 Hive:

Hive is a Hadoop-based data warehouse architecture that was created at Facebook and is used for ad hoc querying using a SQL-like query language as well as more sophisticated analysis. Tables and columns are defined by the users. These tables are used to store and retrieve data. To generate summaries, reports, and analytics, Hive QL, a SQL-like query language, is utilized. Map Reduce jobs are launched by Hive queries. Hive is built for batch processing rather than online transaction processing, and unlike HBase (see below), it does not support real-time queries.

1.3 Pig:

Pig is a high-level data-flow language and execution framework whose compiler generates Map/Reduce program sequences for Hadoop execution. Pig is a program for batch data processing. Pig's infrastructure layer consists of a compiler that converts Pig Latin programs into Map Reduce program sequences. Pig is a Java client-side program that customers install locally the Hadoop cluster is unaffected. Pig's interactive shell is Grunt **[5].**

1.4 Mahout and other expansions to Hadoop programming capabilities:

Hadoop isn't only for processing huge amounts of data. Mahout is an Apache project for creating scalable machine learning libraries, with the majority of the algorithms being based on Hadoop. Clustering, classification, data mining (frequent itemset), and evolutionary programming are some of Mahout's current algorithm emphasis areas. The Mahout clustering and classifier algorithms have obvious applications in bioinformatics, such as clustering huge gene expression data sets and using classifiers to identify biomarkers. In terms of clustering, we should mention that M. Ngazimbi and K. Heafield at Google, among others, have looked at Hadoop MapReduce-based clustering. The “R and Hadoop Integrated Processing Environment” (RHIPE), S. Guhi's Java program that connects the R environment with Hadoop so that MapReduce algorithms may be coded in R, may be of interest to the many bioinformaticians who use R. Pydoop, a Python MapReduce and HDFS API for Hadoop that enables entire MapReduce applications to be built in Python, is now accessible for the expanding community of Python users in biology. These are just a few examples of the huge number of people working on Hadoop extensions. In this limited area, one last example: Clojure, a new programming language that is primarily a functional language (e.g., a version of Lisp that targets the Java Virtual Machine), has been provided a library to assist in the creation of Hadoop tasks **[6].**

1.5 Cascading:

Cascading is a Hadoop project that provides a programming API for designing and running fault-tolerant data processing processes. Cascading is a lightweight Java library that sits on top of Hadoop's MapReduce layer. Cascading is a query processing API that enables programmers to work at a higher level than MapReduce, allowing them to more rapidly build and plan complicated distributed processes based on dependencies **[7].**

1.6 H Base:

Finally, HBase, which is based on Google's Big Table database, is a significant Apache Hadoop-based project. Built on top of the HDFS file system, HBase provides a distributed, fault-tolerant, scalable database with random real-time read/write access to data. Each HBase database is saved as a multidimensional sparse map of rows and columns, with a time stamp in each cell. HBase has its own Java client API, and via TableInput/TableOutputFormat, tables in HBase may be utilized as both an input source and an output target for MapReduce tasks. There is no single point of failure in HBase. HBase manages partial failures using Zookeeper, another Hadoop subproject. The main key is used to access all tables. Additional index tables may be used to create secondary indexes; programmers must de normalize and duplicate. In HBase's basic version, there is no SQL query language. However, there is a Hive/HBase integration project that enables Hive QL expressions to read and write data into HBase databases. There's also the separate HBql project, which adds a SQL dialect and JDBC connectors for HBase. Regions are the components of a table. Each region has a startKey and an endKey, may reside on a separate node, and is made up of multiple HDFS files and blocks that are all duplicated by Hadoop. Only the parent column families are specified in a schema, thus columns may be added to tables on the fly. Each cell is labelled with the column family and column name so that programs can always tell what kind of data item is in that cell. We may notice the simplicity of integrating diverse data sources into a small number of HBase tables for creating a data workspace, with different columns potentially created (on-the-fly) for different rows in the same database, in addition to being able to grow to petabyte size data sets. It is also necessary to have such a facility. (For more on biological integration, see the section below.) Other scalable random access databases, in addition to HBase, are now accessible. HadoopDB is a combination of MapReduce with a traditional relational database system. HadoopDB's database layer is PostgreSQL (one PostgreSQL instance per data chunk per node), the communication layer is Hadoop, and the translation layer is an enhanced version of Hive. There are other non-Hadoop scalable alternatives, such as hyper table and Cassandra that are based on the Google Big Table idea. Other so-called NoSQL scalable databases that may be of interest include Project Voldemort, Dynamo, and Tokyo Tyrant, among others. These non-Hadoop and non-Big Table database systems, on the other hand, are beyond the scope of this article **[8].**

1.7 Use of Hadoop and HBase in Bioinformatics:

1.7.1 Use in next-generation sequencing:

For SNP identification and genotyping, the Cloudburst program links next-generation short read sequencing data to a reference genome. Michael C. Schatz of the University of Maryland developed Cloudburst. In May 2009, Schatz's Cloudburst article placed Hadoop "on the map" in bioinformatics. Following the publication of Cloudburst, Schatz and colleagues at the University of Maryland and Johns Hopkins University (e.g., B. Langmead) created a set of algorithms for analyzing next-generation sequencing data using Hadoop**:**

1) For whole genome resequencing analysis and SNP genotyping from short reads, Crossbow relies on Hadoop.

2) Contrail scales up de Brujin graph building by utilizing Hadoop for de novo assembly from short sequencing data (without needing a reference genome).

3) Myrna utilizes R/Bioconductor to calculate differential gene expression from huge RNA-seq data sets, as well as Bowtie, another UMD tool for rapid short read alignment. Myrna utilizes Hadoop when operating in a cluster. Myrna may also be used on the cloud using Amazon Elastic Map Reduce.

1.7.2 Cloud computing results:

Amazon Elastic Compute Cloud (EC2) and Amazon Elastic MapReduce are cloud computing services that offer scalability. They provide Hadoop, among other batch processing tools. Myrna was built to work on both Elastic Map Reduce and on a local Hadoop cluster. Langmead et al. clearly think that cloud computing is an useful computing architecture, as shown by the fact that they publish their findings in. Schatz has also tried Crossbow on EC2 and thinks that it may be very cost efficient to operate on EC2. (Researchers may also use non-commercial services such as the IBM/Google Cloud Computing Initiative.) In addition, Indiana University (IU) researchers compared MPI, Dryad, Azure (Microsoft), and Hadoop MapReduce, evaluating relative performance using three bioinformatics applications. Judy Qui of Indiana University summarized this work at BOSC 2010. The IU testing shows that clouds and MapReduce have a lot of flexibility, implying that “they will become favored approaches.

1.7.3 Use in other bioinformatics domains:

Hadoop and HBase have been used in bioinformatics applications other than next-generation sequencing. M. Gaggero and colleagues from the Center for Advanced Studies, Research and Development in Sardinia's Distributed Computing Group have published a paper on implementing BLAST and Gene Set Enrichment Analysis (GSEA) in Hadoop. To create an executable mapper for BLAST, a Python wrapper for the NCBI C++ Toolkit and Hadoop Streaming were used. For the MapReduce version, GSEA was developed utilizing rewritten Python routines and Hadoop Streaming. They're now working on Biodoop, a Hadoop-based suite of parallel bioinformatics applications that includes three qualitatively distinct algorithms: BLAST, GSEA, and GRAMMAR. They call their findings "extremely promising," describing MapReduce as a versatile architecture.

1.8 Use in scientific cloud computing, biological data integration and knowledgebase construction:

The Magellan project [67], a collaborative research effort of the National Energy Research Scientific Computing Center (NERSC), Lawrence Berkeley National Laboratory, and the Leadership Computing Facility at Argonne National Laboratory, is looking into scientific cloud computing (ANL). At NERSC, Hadoop and HBase were deployed on a cluster (40 nodes designated for Hadoop, shortly to double), and BLAST calculations were performed using Hadoop in Streaming mode. On the Hadoop nodes, NERSC is investigating the usage of solid state (flash) storage. The DOE Joint Genome Institute has also used Hadoop to extend contigs on the NERSC cluster. In late 2010, the Hadoop cluster at ANL, which is now undergoing testing, will be accessible to researchers. Fill out the Magellan Cloud Computing statement of interest form if you're interested in utilizing clouds for your study. We want to build a scientific data management system that can scale into the petabyte range, store data acquired from our various instruments accurately and reliably, and store the output of analysis software and relevant metadata at the Environmental Molecular Sciences Laboratory, a national user facility at DOE's Pacific Northwest National Laboratory (PNNL). Work on a prototype data repository, i.e., a workspace for integrating high-throughput transcriptomics and proteomics data, began in August 2010 as a pilot project for such an endeavour. This database will be able to hold massive quantities of data from both mass spectrometry-based proteomics studies and next-generation high-throughput sequencing technologies. The prototype database is being built by the author (RCT) on a 25-node cluster utilizing Hadoop and HBase as the framework. We may consider utilizing Hadoop and HBase for the construction of big knowledge bases running on a cluster across the distributed file system in addition to such data warehousing / data integration activities. The US Department of Energy is funding research into large biological knowledge bases, and Kandinsky, a 68-node, 1088-core Linux cluster (64 GB RAM, 8TB disk per node) running Hadoop (Cloudera distribution, under CentOS 5) and HBase, was set up as an exploratory environment at Oak Ridge National Laboratory in 2010. Cloudburst has been deployed as a prototype Hadoop-based application, and the cluster is available for researchers to use for early work on knowledgebase creation and grant proposal assistance.

2. DISCUSSION

Since its release in late 2006, Hadoop has been the best solution for big data processing and storage. Hadoop data processing is based on a master-slave model, which divides a large file job into several small files so that they can be processed separately. This technique was used instead of pushing one large file into a costly super machine to extract some useful information. Hadoop performs well with huge files of big data, but when dealing with tiny files of big data, it may have performance issues such as processing slowdowns, data access delays, excessive latency, and even cluster shutting down. In this article, we will focus on one of Hadoop's constraints that has an impact on data processing speed. One of these restrictions, known as "big data in tiny files," arises when a large number of small files are pushed into a Hadoop cluster, causing the cluster to completely shut down. This paper discusses several aspects of Hadoop, MapReduce and Hive.

3. CONCLUSION

As shown by the huge number of applications listed above, Hadoop and its related open source projects have a broad and expanding bioinformatics community of both users and developers. Researcher provides a concluding point based on early work for the Hadoop/HBase-based PNNL project. That is, for most bioinformatics work, the simplicity of integrating and analysing multiple big, diverse data sources into one data warehouse under Hadoop, in relatively few HBase tables, is just as essential as the scalability provided by Hadoop and HBase.The new generations of mobile devices have high processing power and storage, but they lag behind in terms of software systems for big data storage and processing. Hadoop is a scalable platform that provides distributed storage and computational capabilities on clusters of commodity hardware. Building Hadoop on a mobile network enables the devices to run data intensive computing applications without direct knowledge of underlying distributed systems complexities.

REFERENCES:

1. Diaconita V, Bologa AR, Bologa R Hadoop oriented smart cities architecture. Sensors (Switzerland), 2018, doi: 10.3390/s18041181.

2. O’Driscoll A, Daugelaite J, Sleator RD. Big data’, Hadoop and cloud computing in genomics. Journal of Biomedical Informatics. 2013;46(5):774–781. doi: 10.1016/j.jbi.2013.07.001.

3. Hodge VJ, O’Keefe S, Austin J. Hadoop neural network for parallel and distributed feature selection. Neural Networks, 2016 Jun;78:24-35. doi: 10.1016/j.neunet.2015.08.011.

4. Taylor RC. An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. BMC Bioinformatics, 2010;11(Suppl 12):S1. doi: 10.1186/1471-2105-11-S12-S1.

5. White T. Hadoop: The definitive guide 4th Edition. Online, 2012, doi: citeulike-article-id:4882841.

6. Sirisha N, Kiran KVD. Authorization of data in Hadoop using Apache Sentry. Int. J. Eng. Technol., 2018;7(3.6): 234-236. doi: 10.14419/ijet.v7i3.6.14978.

7. Niemenmaa M, Kallio A, Schumacher A, Klemelä P, Korpelainen E, Heljanko K. Hadoop-BAM: Directly manipulating next generation sequencing data in the cloud. Bioinformatics, 2012 Mar 15;28(6):876-7. doi: 10.1093/bioinformatics/bts054.

8. Polato I, Ré R, Goldman A, Kon F. A comprehensive view of Hadoop research - A systematic literature review. Journal of Network and Computer Applications. 2014;46:1–25. doi: 10.1016/j.jnca.2014.07.022.