Chapter No.1
"Getting Ready to Use R and Hadoop"
In this package, you will find:
A Biography of the author of the book
A preview chapter from the book, Chapter NO.1 "Getting Ready to Use R and Hadoop"
A synopsis of the book’s content
Information on where to buy this book

About the Author

Vignesh Prajapati, from India, is a Big Data enthusiast, a Pingax consultant and a software professional at Enjay. He is an experienced ML Data engineer. He is experienced with Machine learning and Big Data technologies such as R, Hadoop, Mahout, Pig, Hive, and related Hadoop components to analyse datasets to achieve informative insights by data analytics cycles.

He pursued B.E from Gujarat Technological University in 2012 and started his career as Data Engineer at Tatvic. His professional experience includes working on the development of various Data analytics algorithms for Google Analytics data source, for providing economic value to the products. To get the ML in action, he implemented several analytical apps in collaboration with Google Analytics and Google Prediction API services. He also contributes to the R community by developing the RGoogleAnalytics' R library as an open source code Google project and writes articles on Data-driven technologies.

Vignesh is not limited to a single domain; he has also worked for developing various interactive apps via various Google APIs, such as Google Analytics API, Realtime API, Google Prediction API, Google Chart API, and Translate API with the Java and PHP platforms. He is highly interested in the development of open source technologies.

For More Information:
Vignesh has also reviewed the Apache Mahout Cookbook for Packt Publishing. This book provides a fresh, scope-oriented approach to the Mahout world for beginners as well as advanced users. Mahout Cookbook is specially designed to make users aware of the different possible machine learning applications, strategies, and algorithms to produce an intelligent as well as Big Data application.

For More Information:
Big Data Analytics with R and Hadoop

The volume of data that enterprises acquire every day is increasing exponentially. It is now possible to store these vast amounts of information on low cost platforms such as Hadoop.

The conundrum these organizations now face is what to do with all this data and how to glean key insights from this data. Thus R comes into picture. R is a very amazing tool that makes it a snap to run advanced statistical models on data, translate the derived models into colorful graphs and visualizations, and do a lot more functions related to data science.

One key drawback of R, though, is that it is not very scalable. The core R engine can process and work on very limited amount of data. As Hadoop is very popular for Big Data processing, corresponding R with Hadoop for scalability is the next logical step.

This book is dedicated to R and Hadoop and the intricacies of how data analytics operations of R can be made scalable by using a platform as Hadoop.

With this agenda in mind, this book will cater to a wide audience including data scientists, statisticians, data architects, and engineers who are looking for solutions to process and analyze vast amounts of information using R and Hadoop.

Using R with Hadoop will provide an elastic data analytics platform that will scale depending on the size of the dataset to be analyzed. Experienced programmers can then write Map/Reduce modules in R and run it using Hadoop's parallel processing Map/Reduce mechanism to identify patterns in the dataset.

Introducing R

R is an open source software package to perform statistical analysis on data. R is a programming language used by data scientist statisticians and others who need to make statistical analysis of data and glean key insights from data using mechanisms, such as regression, clustering, classification, and text analysis. R is registered under GNU (General Public License). It was developed by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, which is currently handled by the R Development Core Team. It can be considered as a different implementation of S, developed by Johan Chambers at Bell Labs. There are some important differences, but a lot of the code written in S can be unaltered using the R interpreter engine.

R provides a wide variety of statistical, machine learning (linear and nonlinear modeling, classic statistical tests, time-series analysis, classification, clustering) and graphical techniques, and is highly extensible. R has various built-in as well as extended functions for statistical, machine learning, and visualization tasks such as:

For More Information:
- Data extraction
- Data cleaning
- Data loading
- Data transformation
- Statistical analysis
- Predictive modeling
- Data visualization

It is one of the most popular open source statistical analysis packages available on the market today. It is crossplatform, has a very wide community support, and a large and ever-growing user community who are adding new packages every day. With its growing list of packages, R can now connect with other data stores, such as MySQL, SQLite, MongoDB, and Hadoop for data storage activities.

**Understanding features of R**

Let's see different useful features of R:

- Effective programming language
- Relational database support
- Data analytics
- Data visualization
- Extension through the vast library of R packages

**Studying the popularity of R**

The graph provided from KD suggests that R is the most popular language for data analysis and mining:

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>Votes</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>257</td>
<td>45%</td>
</tr>
<tr>
<td>SQL</td>
<td>184</td>
<td>32%</td>
</tr>
<tr>
<td>Python</td>
<td>140</td>
<td>25%</td>
</tr>
<tr>
<td>Java</td>
<td>139</td>
<td>24%</td>
</tr>
<tr>
<td>SAS</td>
<td>121</td>
<td>21%</td>
</tr>
<tr>
<td>MATLAB</td>
<td>83</td>
<td>15%</td>
</tr>
<tr>
<td>C/C++</td>
<td>73</td>
<td>13%</td>
</tr>
<tr>
<td>Unix shell/awk/gawk/</td>
<td>59</td>
<td>10%</td>
</tr>
<tr>
<td>Perl</td>
<td>45</td>
<td>7.9%</td>
</tr>
<tr>
<td>Hadoop/Pig/Hive</td>
<td>35</td>
<td>6.1%</td>
</tr>
<tr>
<td>Lisp</td>
<td>4</td>
<td>0.7%</td>
</tr>
<tr>
<td>Other</td>
<td>70</td>
<td>12.0%</td>
</tr>
<tr>
<td>None</td>
<td>7</td>
<td>1.2%</td>
</tr>
</tbody>
</table>

For More Information:
The following graph provides details about the total number of R packages released by R users from 2005 to 2013. This is how we explore R users. The growth was exponential in 2012 and it seems that 2013 is on track to beat that. R allows performing Data analytics by various statistical and machine learning operations as follows:

- Regression
- Classification
- Clustering
- Recommendation
- Text mining

Introducing Big Data

Big Data has to deal with large and complex datasets that can be structured, semi-structured, or unstructured and will typically not fit into memory to be processed. They have to be processed in place, which means that computation has to be done where the data resides for processing. When we talk to developers, the people actually building Big Data systems and applications, we get a better idea of what they mean about 3Vs. They typically would mention the 3Vs model of Big Data, which are velocity, volume, and variety.

For More Information:
Velocity refers to the low latency, real-time speed at which the analytics need to be applied. A typical example of this would be to perform analytics on a continuous stream of data originating from a social networking site or aggregation of disparate sources of data.

Volume refers to the size of the dataset. It may be in KB, MB, GB, TB, or PB based on the type of the application that generates or receives the data.

Variety refers to the various types of the data that can exist, for example, text, audio, video, and photos.

Big Data usually includes datasets with sizes. It is not possible for such systems to process this amount of data within the time frame mandated by the business. Big Data volumes are a constantly moving target, as of 2012 ranging from a few dozen terabytes to many petabytes of data in a single dataset. Faced with this seemingly insurmountable challenge, entirely new platforms are called Big Data platforms.

Getting information about popular organizations that hold Big Data

Some of the popular organizations that hold Big Data are as follows:

- Facebook: It has 40 PB of data and captures 100 TB/day
- Yahoo!: It has 60 PB of data
- Twitter: It captures 8 TB/day
- EBay: It has 40 PB of data and captures 50 TB/day

For More Information:  
How much data is considered as Big Data differs from company to company. Though true that one company's Big Data is another's small, there is something common: doesn't fit in memory, nor disk, has rapid influx of data that needs to be processed and would benefit from distributed software stacks. For some companies, 10 TB of data would be considered Big Data and for others 1 PB would be Big Data.

So only you can determine whether the data is really Big Data. It is sufficient to say that it would start in the low terabyte range.

Also, a question well worth asking is, as you are not capturing and retaining enough of your data do you think you do not have a Big Data problem now? In some scenarios, companies literally discard data, because there wasn't a cost effective way to store and process it. With platforms as Hadoop, it is possible to start capturing and storing all that data.

**Introducing Hadoop**

Apache Hadoop is an open source Java framework for processing and querying vast amounts of data on large clusters of commodity hardware. Hadoop is a top level Apache project, initiated and led by Yahoo! and Doug Cutting. It relies on an active community of contributors from all over the world for its success.

With a significant technology investment by Yahoo!, Apache Hadoop has become an enterprise-ready cloud computing technology. It is becoming the industry de facto framework for Big Data processing.

Hadoop changes the economics and the dynamics of large-scale computing. Its impact can be boiled down to four salient characteristics. Hadoop enables scalable, cost-effective, flexible, fault-tolerant solutions.

**Exploring Hadoop features**

Apache Hadoop has two main features:

- HDFS (Hadoop Distributed File System)
- MapReduce

For More Information:  
Studying Hadoop components

Hadoop includes an ecosystem of other products built over the core HDFS and MapReduce layer to enable various types of operations on the platform. A few popular Hadoop components are as follows:

- **Mahout**: This is an extensive library of machine learning algorithms.
- **Pig**: Pig is a high-level language (such as PERL) to analyze large datasets with its own language syntax for expressing data analysis programs, coupled with infrastructure for evaluating these programs.
- **Hive**: Hive is a data warehouse system for Hadoop that facilitates easy data summarization, ad hoc queries, and the analysis of large datasets stored in HDFS. It has its own SQL-like query language called **Hive Query Language (HQL)**, which is used to issue query commands to Hadoop.
- **HBase**: HBase (Hadoop Database) is a distributed, column-oriented database. HBase uses HDFS for the underlying storage. It supports both batch style computations using MapReduce and atomic queries (random reads).
- **Sqoop**: Apache Sqoop is a tool designed for efficiently transferring bulk data between Hadoop and Structured Relational Databases. **Sqoop** is an abbreviation for (SQ) to Ha(op).
- **ZooKeeper**: ZooKeeper is a centralized service to maintain configuration information, naming, providing distributed synchronization, and group services, which are very useful for a variety of distributed systems.
- **Ambari**: A web-based tool for provisioning, managing, and monitoring Apache Hadoop clusters, which includes support for Hadoop HDFS, Hadoop MapReduce, Hive, HCatalog, HBase, ZooKeeper, Oozie, Pig, and Sqoop.

Understanding the reason for using R and Hadoop together

I would also say that sometimes the data resides on the HDFS (in various formats). Since a lot of data analysts are very productive in R, it is natural to use R to compute with the data stored through Hadoop-related tools.

As mentioned earlier, the strengths of R lie in its ability to analyze data using a rich library of packages but fall short when it comes to working on very large datasets. The strength of Hadoop on the other hand is to store and process very large amounts of data in the TB and even PB range. Such vast datasets cannot be processed in memory as the RAM of each machine cannot hold such large datasets. The options would be to run analysis on limited chunks also known as sampling or to correspond the analytical power of R with the storage and processing power of Hadoop and you arrive at an ideal solution. Such solutions can also be achieved in the cloud using platforms such as Amazon EMR.
What This Book Covers

Chapter 1, Getting Ready to Use R and Hadoop, gives an introduction as well as the process of installing R and Hadoop.

Chapter 2, Writing Hadoop MapReduce Programs, covers basics of Hadoop MapReduce and ways to execute MapReduce using Hadoop.

Chapter 3, Integrating R and Hadoop, shows deployment and running of sample MapReduce programs for RHadoop and RHIPE by various data handling processes.

Chapter 4, Using Hadoop Streaming with R, shows how to use Hadoop Streaming with R.

Chapter 5, Learning Data Analytics with R and Hadoop, introduces the Data analytics project life cycle by demonstrating with real-world Data analytics problems.

Chapter 6, Understanding Big Data Analysis with Machine Learning, covers performing Big Data analytics by machine learning techniques with RHadoop.

Chapter 7, Importing and Exporting Data from Various DBs, covers how to interface with popular relational databases to import and export data operations with R.

Appendix, References, describes links to additional resources regarding the content of all the chapters being present.

The first chapter has been bundled with several topics on R and Hadoop basics as follows:

- R Installation, features, and data modeling
- Hadoop installation, features, and components

In the preface, we introduced you to R and Hadoop. This chapter will focus on getting you up and running with these two technologies. Until now, R has been used mainly for statistical analysis, but due to the increasing number of functions and packages, it has become popular in several fields, such as machine learning, visualization, and data operations. R will not load all data (Big Data) into machine memory. So, Hadoop can be chosen to load the data as Big Data. Not all algorithms work across Hadoop, and the algorithms are, in general, not R algorithms. Despite this, analytics with R have several issues related to large data. In order to analyze the dataset, R loads it into the memory, and if the dataset is large, it will fail with exceptions such as "cannot allocate vector of size x". Hence, in order to process large datasets, the processing power of R can be vastly magnified by combining it with the power of a Hadoop cluster. Hadoop is very a popular framework that provides such parallel processing capabilities. So, we can use R algorithms or analysis processing over Hadoop clusters to get the work done.

For More Information:
If we think about a combined RHadoop system, R will take care of data analysis operations with the preliminary functions, such as data loading, exploration, analysis, and visualization, and Hadoop will take care of parallel data storage as well as computation power against distributed data.

Prior to the advent of affordable Big Data technologies, analysis used to be run on limited datasets on a single machine. Advanced machine learning algorithms are very effective when applied to large datasets, and this is possible only with large clusters where data can be stored and processed with distributed data storage systems. In the next section, we will see how R and Hadoop can be installed on different operating systems and the possible ways to link R and Hadoop.

### Installing R

You can download the appropriate version by visiting the official R website.

Here are the steps provided for three different operating systems. We have considered Windows, Linux, and Mac OS for R installation. Download the latest version of R as it will have all the latest patches and resolutions to the past bugs.

For Windows, follow the given steps:

1. Navigate to [www.r-project.org](http://www.r-project.org).
2. Click on the CRAN section, select **CRAN mirror**, and select your Windows OS (stick to Linux; Hadoop is almost always used in a Linux environment).
3. Download the latest R version from the mirror.
4. Execute the downloaded .exe to install R.

For Linux-Ubuntu, follow the given steps:

1. Navigate to [www.r-project.org](http://www.r-project.org).
2. Click on the CRAN section, select **CRAN mirror**, and select your OS.
3. In the /etc/apt/sources.list file, add the CRAN `<mirror>` entry.
4. Download and update the package lists from the repositories using the `sudo apt-get update` command.
5. Install R system using the `sudo apt-get install r-base` command.

For More Information:

For Linux-RHEL/CentOS, follow the given steps:

1. Navigate to www.r-project.org.
2. Click on CRAN, select CRAN mirror, and select Red Hat OS.
3. Download the R-*.core-*.rpm file.
4. Install the .rpm package using the `rpm -ivh R-*.core-*.rpm` command.
5. Install R system using `sudo yum install R`.

For Mac, follow the given steps:

1. Navigate to www.r-project.org.
2. Click on CRAN, select CRAN mirror, and select your OS.
3. Download the following files: pkg, gfortran-*.dmg, and tcltk-*.dmg.
4. Install the R-*.pkg file.
5. Then, install the gfortran-*.dmg and tcltk-*.dmg files.

After installing the base R package, it is advisable to install RStudio, which is a powerful and intuitive Integrated Development Environment (IDE) for R.

We can use R distribution of Revolution Analytics as a Modern Data analytics tool for statistical computing and predictive analytics, which is available in free as well as premium versions. Hadoop integration is also available to perform Big Data analytics.

**Installing RStudio**

To install RStudio, perform the following steps:

2. Download the latest version of RStudio for your operating system.
3. Execute the installer file and install RStudio.

The RStudio organization and user community has developed a lot of R packages for graphics and visualization, such as ggplot2, plyr, Shiny, Rpubs, and devtools.

For More Information:  
Understanding the features of R language
There are over 3,000 R packages and the list is growing day by day. It would be beyond the scope of any book to even attempt to explain all these packages. This book focuses only on the key features of R and the most frequently used and popular packages.

Using R packages
R packages are self-contained units of R functionality that can be invoked as functions. A good analogy would be a .jar file in Java. There is a vast library of R packages available for a very wide range of operations ranging from statistical operations and machine learning to rich graphic visualization and plotting. Every package will consist of one or more R functions. An R package is a re-usable entity that can be shared and used by others. R users can install the package that contains the functionality they are looking for and start calling the functions in the package. A comprehensive list of these packages can be found at http://cran.r-project.org/ called Comprehensive R Archive Network (CRAN).

Performing data operations
R enables a wide range of operations. Statistical operations, such as mean, min, max, probability, distribution, and regression. Machine learning operations, such as linear regression, logistic regression, classification, and clustering. Universal data processing operations are as follows:

• **Data cleaning**: This option is to clean massive datasets
• **Data exploration**: This option is to explore all the possible values of datasets
• **Data analysis**: This option is to perform analytics on data with descriptive and predictive analytics data visualization, that is, visualization of analysis output programming

To build an effective analytics application, sometimes we need to use the online Application Programming Interface (API) to dig up the data, analyze it with expedient services, and visualize it by third-party services. Also, to automate the data analysis process, programming will be the most useful feature to deal with.
R has its own programming language to operate data. Also, the available package can help to integrate R with other programming features. R supports object-oriented programming concepts. It is also capable of integrating with other programming languages, such as Java, PHP, C, and C++. There are several packages that will act as middle-layer programming features to aid in data analytics, which are similar to sqldf, httr, RMongo, RgoogleMaps, RGoogleAnalytics, and google-prediction-api-r-client.

Increasing community support
As the number of R users are escalating, the groups related to R are also increasing. So, R learners or developers can easily connect and get their uncertainty solved with the help of several R groups or communities.

The following are many popular sources that can be found useful:

- **R mailing list**: This is an official R group created by R project owners.
- **R blogs**: R has countless bloggers who are writing on several R applications. One of the most popular blog websites is http://www.r-bloggers.com/ where all the bloggers contribute their blogs.
- **Stack overflow**: This is a great technical knowledge sharing platform where the programmers can post their technical queries and enthusiast programmers suggest a solution. For more information, visit http://stats.stackexchange.com/.
- **Groups**: There are many other groups existing on LinkedIn and Meetup where professionals across the world meet to discuss their problems and innovative ideas.
Performing data modeling in R

Data modeling is a machine learning technique to identify the hidden pattern from the historical dataset, and this pattern will help in future value prediction over the same data. This technique highly focuses on past user actions and learns their taste. Most of these data modeling techniques have been adopted by many popular organizations to understand the behavior of their customers based on their past transactions. These techniques will analyze data and predict for the customers what they are looking for. Amazon, Google, Facebook, eBay, LinkedIn, Twitter, and many other organizations are using data mining for changing the definition applications.

The most common data mining techniques are as follows:

- **Regression**: In statistics, regression is a classic technique to identify the scalar relationship between two or more variables by fitting the state line on the variable values. That relationship will help to predict the variable value for future events. For example, any variable $y$ can be modeled as linear function of another variable $x$ with the formula $y = mx+c$. Here, $x$ is the predictor variable, $y$ is the response variable, $m$ is slope of the line, and $c$ is the intercept. Sales forecasting of products or services and predicting the price of stocks can be achieved through this regression. R provides this regression feature via the `lm` method, which is by default present in R.

- **Classification**: This is a machine-learning technique used for labeling the set of observations provided for training examples. With this, we can classify the observations into one or more labels. The likelihood of sales, online fraud detection, and cancer classification (for medical science) are common applications of classification problems. Google Mail uses this technique to classify e-mails as spam or not. Classification features can be served by `glm`, `glmnet`, `ksvm`, `svm`, and `randomForest` in R.

- **Clustering**: This technique is all about organizing similar items into groups from the given collection of items. User segmentation and image compression are the most common applications of clustering. Market segmentation, social network analysis, organizing the computer clustering, and astronomical data analysis are applications of clustering. Google News uses these techniques to group similar news items into the same category. Clustering can be achieved through the `knn`, `kmeans`, `dist`, `pvclust`, and `Mclust` methods in R.
• **Recommendation**: The recommendation algorithms are used in recommender systems where these systems are the most immediately recognizable machine learning techniques in use today. Web content recommendations may include similar websites, blogs, videos, or related content. Also, recommendation of online items can be helpful for cross-selling and up-selling. We have all seen online shopping portals that attempt to recommend books, mobiles, or any items that can be sold on the Web based on the user's past behavior. Amazon is a well-known e-commerce portal that generates 29 percent of sales through recommendation systems. Recommender systems can be implemented via `Recommender()` with the `recommenderlab` package in R.

### Installing Hadoop

Now, we presume that you are aware of R, what it is, how to install it, what it's key features are, and why you may want to use it. Now we need to know the limitations of R (this is a better introduction to Hadoop). Before processing the data; R needs to load the data into **random access memory** (RAM). So, the data needs to be smaller than the available machine memory. For data that is larger than the machine memory, we consider it as Big Data (only in our case as there are many other definitions of Big Data).

To avoid this Big Data issue, we need to scale the hardware configuration; however, this is a temporary solution. To get this solved, we need to get a Hadoop cluster that is able to store it and perform parallel computation across a large computer cluster. Hadoop is the most popular solution. Hadoop is an open source Java framework, which is the top level project handled by the Apache software foundation. Hadoop is inspired by the Google filesystem and MapReduce, mainly designed for operating on Big Data by distributed processing.

Hadoop mainly supports Linux operating systems. To run this on Windows, we need to use VMWare to host Ubuntu within the Windows OS. There are many ways to use and install Hadoop, but here we will consider the way that supports R best. Before we combine R and Hadoop, let us understand what Hadoop is.

Machine learning contains all the data modeling techniques that can be explored with the web link [http://en.wikipedia.org/wiki/Machine_learning](http://en.wikipedia.org/wiki/Machine_learning).

Getting Ready to Use R and Hadoop

Understanding different Hadoop modes
Hadoop is used with three different modes:

- **The standalone mode**: In this mode, you do not need to start any Hadoop daemons. Instead, just call `~/Hadoop-directory/bin/hadoop` that will execute a Hadoop operation as a single Java process. This is recommended for testing purposes. This is the default mode and you don’t need to configure anything else. All daemons, such as NameNode, DataNode, JobTracker, and TaskTracker run in a single Java process.

- **The pseudo mode**: In this mode, you configure Hadoop for all the nodes. A separate Java Virtual Machine (JVM) is spawned for each of the Hadoop components or daemons like mini cluster on a single host.

- **The full distributed mode**: In this mode, Hadoop is distributed across multiple machines. Dedicated hosts are configured for Hadoop components. Therefore, separate JVM processes are present for all daemons.

Understanding Hadoop installation steps
Hadoop can be installed in several ways; we will consider the way that is better to integrate with R. We will choose Ubuntu OS as it is easy to install and access it.

1. Installing Hadoop on Linux, Ubuntu flavor (single and multinode cluster).
2. Installing Cloudera Hadoop on Ubuntu.

Installing Hadoop on Linux, Ubuntu flavor (single node cluster)
To install Hadoop over Ubuntu OS with the pseudo mode, we need to meet the following prerequisites:

- Sun Java 6
- Dedicated Hadoop system user
- Configuring SSH
- Disabling IPv6

The provided Hadoop installation will be supported with Hadoop MRv1.

For More Information:
Follow the given steps to install Hadoop:

1. Download the latest Hadoop sources from the Apache software foundation. Here we have considered Apache Hadoop 1.0.3, whereas the latest version is 1.1.x.

   // Locate to Hadoop installation directory
   $ cd /usr/local

   // Extract the tar file of Hadoop distribution
   $ sudo tar xzf hadoop-1.0.3.tar.gz

   // To move Hadoop resources to hadoop folder
   $ sudo mv hadoop-1.0.3 hadoop

   // Make user-hduser from group-hadoop as owner of hadoop directory
   $ sudo chown -R hduser:hadoop hadoop

2. Add the $JAVA_HOME and $HADOOP_HOME variables to the .bashrc file of Hadoop system user and the updated .bashrc file looks as follows:

   // Setting the environment variables for running Java and Hadoop commands
   export HADOOP_HOME=/usr/local/hadoop
   export JAVA_HOME=/usr/lib/jvm/java-6-sun

   // alias for Hadoop commands
   unalias fs &> /dev/null
   alias fs="hadoop fs"
   unalias hls &> /dev/null
   alias hls="fs -ls"

   // Defining the function for compressing the MapReduce job output by lzop command
   lzohead () {
     hadoopfs -cat $1 | lzop -dc | head -1000 | less
   }

   // Adding Hadoop_HoME variable to PATH
   export PATH=$PATH:$HADOOP_HOME/bin

3. Update the Hadoop configuration files with the conf/*-site.xml format.
Finally, the three files will look as follows:

- **conf/core-site.xml:**
  
  ```xml
  <property>
  <name>hadoop.tmp.dir</name>
  <value>/app/hadoop/tmp</value>
  <description>A base for other temporary directories.</description>
  </property>

  <property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:54310</value>
  <description>The name of the default filesystem. A URI whose scheme and authority determine the FileSystem implementation. The uri's scheme determines the config property (fs.SCHEME.impl) naming the FileSystem implementation class. The uri's authority is used to determine the host, port, etc. for a filesystem.</description>
  </property>

- **conf/mapred-site.xml:**
  
  ```xml
  <property>
  <name>mapred.job.tracker</name>
  <value>localhost:54311</value>
  <description>The host and port that the MapReduce job tracker runs at. If "local", then jobs are run in-process as a single map and reduce task.</description>
  </property>

- **conf/hdfs-site.xml:**
  
  ```xml
  <property>
  <name>dfs.replication</name>
  <value>1</value>
  <description>Default block replication.
  The actual number of replications can be specified when the file is created.
  The default is used if replication is not specified in create time.
  </description>
  ```
After completing the editing of these configuration files, we need to set up the distributed filesystem across the Hadoop clusters or node.

- Format Hadoop Distributed File System (HDFS) via NameNode by using the following command line:
  ```
  hduser@ubuntu:~$ /usr/local/hadoop/bin/hadoopnamenode -format
  ```

- Start your single node cluster by using the following command line:
  ```
  hduser@ubuntu:~$ /usr/local/hadoop/bin/start-all.sh
  ```

### Downloading the example code
You can download the example code files for all Packt books you have purchased from your account at http://www.packtpub.com. If you purchased this book elsewhere, you can visit http://www.packtpub.com/support and register to have the files e-mailed directly to you.

### Installing Hadoop on Linux, Ubuntu flavor (multinode cluster)
We learned how to install Hadoop on a single node cluster. Now we will see how to install Hadoop on a multinode cluster (the full distributed mode).

For this, we need several nodes configured with a single node Hadoop cluster. To install Hadoop on multinodes, we need to have that machine configured with a single node Hadoop cluster as described in the last section.

After getting the single node Hadoop cluster installed, we need to perform the following steps:

1. In the networking phase, we are going to use two nodes for setting up a full distributed Hadoop mode. To communicate with each other, the nodes need to be in the same network in terms of software and hardware configuration.

2. Among these two, one of the nodes will be considered as master and the other will be considered as slave. So, for performing Hadoop operations, master needs to be connected to slave. We will enter `192.168.0.1` in the master machine and `192.168.0.2` in the slave machine.

3. Update the `/etc/hosts` directory in both the nodes. It will look as `192.168.0.1 master` and `192.168.0.2 slave`.

For More Information:  
You can perform the Secure Shell (SSH) setup similar to what we did for a single node cluster setup. For more details, visit http://www.michael-noll.com.

4. Updating conf/*-site.xml: We must change all these configuration files in all of the nodes.
   - conf/core-site.xml and conf/mapred-site.xml: In the single node setup, we have updated these files. So, now we need to just replace localhost by master in the value tag.
   - conf/hdfs-site.xml: In the single node setup, we have set the value of dfs.replication as 1. Now we need to update this as 2.

5. In the formatting HDFS phase, before we start the multinode cluster, we need to format HDFS with the following command (from the master node):

   bin/hadoop namenode -format

Now, we have completed all the steps to install the multinode Hadoop cluster. To start the Hadoop clusters, we need to follow these steps:

1. Start HDFS daemons:
   hduser@master:/usr/local/hadoop$ bin/start-dfs.sh

2. Start MapReduce daemons:
   hduser@master:/usr/local/hadoop$ bin/start-mapred.sh

3. Alternatively, we can start all the daemons with a single command:
   hduser@master:/usr/local/hadoop$ bin/start-all.sh

4. To stop all these daemons, fire:
   hduser@master:/usr/local/hadoop$ bin/stop-all.sh

These installation steps are reproduced after being inspired by the blogs (http://www.michael-noll.com) of Michael Noll, who is a researcher and Software Engineer based in Switzerland, Europe. He works as a Technical lead for a large scale computing infrastructure on the Apache Hadoop stack at VeriSign.

Now the Hadoop cluster has been set up on your machines. For the installation of the same Hadoop cluster on single node or multinode with extended Hadoop components, try the Cloudera tool.
Installing Cloudera Hadoop on Ubuntu

Cloudera Hadoop (CDH) is Cloudera's open source distribution that targets enterprise class deployments of Hadoop technology. Cloudera is also a sponsor of the Apache software foundation. CDH is available in two versions: CDH3 and CDH4. To install one of these, you must have Ubuntu with either 10.04 LTS or 12.04 LTS (also, you can try CentOS, Debian, and Red Hat systems). Cloudera manager will make this installation easier for you if you are installing a Hadoop on cluster of computers, which provides GUI-based Hadoop and its component installation over a whole cluster. This tool is very much recommended for large clusters.

We need to meet the following prerequisites:

- Configuring SSH
- OS with the following criteria:
  - Ubuntu 10.04 LTS or 12.04 LTS with 64 bit
  - Red Hat Enterprise Linux 5 or 6
  - CentOS 5 or 6
  - Oracle Enterprise Linux 5
  - SUSE Linux Enterprise server 11 (SP1 or lasso)
  - Debian 6.0

The installation steps are as follows:

1. Download and run the Cloudera manager installer: To initialize the Cloudera manager installation process, we need to first download the cloudera-manager-installer.bin file from the download section of the Cloudera website. After that, store it at the cluster so that all the nodes can access this. Allow ownership for execution permission of cloudera-manager-installer.bin to the user. Run the following command to start execution.

   $ sudo ./cloudera-manager-installer.bin

2. Read the Cloudera manager Readme and then click on Next.

3. Start the Cloudera manager admin console: The Cloudera manager admin console allows you to use Cloudera manager to install, manage, and monitor Hadoop on your cluster. After accepting the license from the Cloudera service provider, you need to traverse to your local web browser by entering http://localhost:7180 in your address bar. You can also use any of the following browsers:

   - Firefox 11 or higher
   - Google Chrome
   - Internet Explorer
   - Safari

For More Information:
4. Log in to the Cloudera manager console with the default credentials using admin for both the username and password. Later on you can change it as per your choice.

5. Use the Cloudera manager for automated CDH3 installation and configuration via browser: This step will install most of the required Cloudera Hadoop packages from Cloudera to your machines. The steps are as follows:
   1. Install and validate your Cloudera manager license key file if you have chosen a full version of software.
   2. Specify the hostname or IP address range for your CDH cluster installation.
   3. Connect to each host with SSH.
   4. Install the Java Development Kit (JDK) (if not already installed), the Cloudera manager agent, and CDH3 or CDH4 on each cluster host.
   5. Configure Hadoop on each node and start the Hadoop services.

6. After running the wizard and using the Cloudera manager, you should change the default administrator password as soon as possible. To change the administrator password, follow these steps:
   1. Click on the icon with the gear sign to display the administration page.
   2. Open the Password tab.
   3. Enter a new password twice and then click on Update.

7. Test the Cloudera Hadoop installation: You can check the Cloudera manager installation on your cluster by logging into the Cloudera manager admin console and by clicking on the Services tab. You should see something like the following screenshot:

For More Information:  
Cloudera manager admin console

8. You can also click on each service to see more detailed information. For example, if you click on the hdfs1 link, you might see something like the following screenshot:

To avoid these installation steps, use preconfigured Hadoop instances with Amazon Elastic MapReduce and MapReduce.

If you want to use Hadoop on Windows, try the HDP tool by Hortonworks. This is 100 percent open source, enterprise grade distribution of Hadoop. You can download the HDP tool at http://hortonworks.com/download/.

For More Information:
Understanding Hadoop features

Hadoop is specially designed for two core concepts: HDFS and MapReduce. Both are related to distributed computation. MapReduce is believed as the heart of Hadoop that performs parallel processing over distributed data.

Let us see more details on Hadoop's features:

- HDFS
- MapReduce

Understanding HDFS

HDFS is Hadoop's own rack-aware filesystem, which is a UNIX-based data storage layer of Hadoop. HDFS is derived from concepts of Google filesystem. An important characteristic of Hadoop is the partitioning of data and computation across many (thousands of) hosts, and the execution of application computations in parallel, close to their data. On HDFS, data files are replicated as sequences of blocks in the cluster. A Hadoop cluster scales computation capacity, storage capacity, and I/O bandwidth by simply adding commodity servers. HDFS can be accessed from applications in many different ways. Natively, HDFS provides a Java API for applications to use.

The Hadoop clusters at Yahoo! span 40,000 servers and store 40 petabytes of application data, with the largest Hadoop cluster being 4,000 servers. Also, one hundred other organizations worldwide are known to use Hadoop.

Understanding the characteristics of HDFS

Let us now look at the characteristics of HDFS:

- Fault tolerant
- Runs with commodity hardware
- Able to handle large datasets
- Master slave paradigm
- Write once file access only

Understanding MapReduce

MapReduce is a programming model for processing large datasets distributed on a large cluster. MapReduce is the heart of Hadoop. Its programming paradigm allows performing massive data processing across thousands of servers configured with Hadoop clusters. This is derived from Google MapReduce.

For More Information:
Hadoop MapReduce is a software framework for writing applications easily, which process large amounts of data (multiterabyte datasets) in parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. This MapReduce paradigm is divided into two phases, Map and Reduce that mainly deal with key and value pairs of data. The Map and Reduce task run sequentially in a cluster; the output of the Map phase becomes the input for the Reduce phase. These phases are explained as follows:

- **Map phase**: Once divided, datasets are assigned to the task tracker to perform the Map phase. The data functional operation will be performed over the data, emitting the mapped key and value pairs as the output of the Map phase.

- **Reduce phase**: The master node then collects the answers to all the subproblems and combines them in some way to form the output; the answer to the problem it was originally trying to solve.

The five common steps of parallel computing are as follows:

1. **Preparing the Map() input**: This will take the input data row wise and emit key value pairs per rows, or we can explicitly change as per the requirement.
   - Map input: list (k1, v1)

2. **Run the user-provided Map() code**
   - Map output: list (k2, v2)

3. **Shuffle the Map output to the Reduce processors. Also, shuffle the similar keys (grouping them) and input them to the same reducer.**

4. **Run the user-provided Reduce() code**: This phase will run the custom reducer code designed by developer to run on shuffled data and emit key and value.
   - Reduce input: (k2, list(v2))
   - Reduce output: (k3, v3)

5. **Produce the final output**: Finally, the master node collects all reducer output and combines and writes them in a text file.

The reference links to review on Google filesystem can be found at http://research.google.com/archive/gfs.html and Google MapReduce can be found at http://research.google.com/archive/mapreduce.html.

---

For More Information:
Learning the HDFS and MapReduce architecture

Since HDFS and MapReduce are considered to be the two main features of the Hadoop framework, we will focus on them. So, let's first start with HDFS.

Understanding the HDFS architecture

HDFS can be presented as the master/slave architecture. HDFS master is named as NameNode whereas slave as DataNode. NameNode is a server that manages the filesystem namespace and adjusts the access (open, close, rename, and more) to files by the client. It divides the input data into blocks and announces which data block will be store in which DataNode. DataNode is a slave machine that stores the replicas of the partitioned dataset and serves the data as the request comes. It also performs block creation and deletion.

The internal mechanism of HDFS divides the file into one or more blocks; these blocks are stored in a set of data nodes. Under normal circumstances of the replication factor three, the HDFS strategy is to place the first copy on the local node, second copy on the local rack with a different node, and a third copy into different racks with different nodes. As HDFS is designed to support large files, the HDFS block size is defined as 64 MB. If required, this can be increased.

Understanding HDFS components

HDFS is managed with the master-slave architecture included with the following components:

- **NameNode**: This is the master of the HDFS system. It maintains the directories, files, and manages the blocks that are present on the DataNodes.

- **DataNode**: These are slaves that are deployed on each machine and provide actual storage. They are responsible for serving read-and-write data requests for the clients.

- **Secondary NameNode**: This is responsible for performing periodic checkpoints. So, if the NameNode fails at any time, it can be replaced with a snapshot image stored by the secondary NameNode checkpoints.
Understanding the MapReduce architecture

MapReduce is also implemented over master-slave architectures. Classic MapReduce contains job submission, job initialization, task assignment, task execution, progress and status update, and job completion-related activities, which are mainly managed by the JobTracker node and executed by TaskTracker. Client application submits a job to the JobTracker. Then input is divided across the cluster. The JobTracker then calculates the number of map and reducer to be processed. It commands the TaskTracker to start executing the job. Now, the TaskTracker copies the resources to a local machine and launches JVM to map and reduce program over the data. Along with this, the TaskTracker periodically sends update to the JobTracker, which can be considered as the heartbeat that helps to update JobID, job status, and usage of resources.

Understanding MapReduce components

MapReduce is managed with master-slave architecture included with the following components:

- **JobTracker**: This is the master node of the MapReduce system, which manages the jobs and resources in the cluster (TaskTrackers). The JobTracker tries to schedule each map as close to the actual data being processed on the TaskTracker, which is running on the same DataNode as the underlying block.

- **TaskTracker**: These are the slaves that are deployed on each machine. They are responsible for running the map and reducing tasks as instructed by the JobTracker.

Understanding the HDFS and MapReduce architecture by plot

In this plot, both HDFS and MapReduce master and slave components have been included, where NameNode and DataNode are from HDFS and JobTracker and TaskTracker are from the MapReduce paradigm.

For More Information:

Both paradigms consisting of master and slave candidates have their own specific responsibility to handle MapReduce and HDFS operations. In the next plot, there is a plot with two sections: the preceding one is a MapReduce layer and the following one is an HDFS layer.

Hadoop is a top-level Apache project and is a very complicated Java framework. To avoid technical complications, the Hadoop community has developed a number of Java frameworks that has added an extra value to Hadoop features. They are considered as Hadoop subprojects. Here, we are departing to discuss several Hadoop components that can be considered as an abstraction of HDFS or MapReduce.
Understanding Hadoop subprojects

Mahout is a popular data mining library. It takes the most popular data mining scalable machine learning algorithms for performing clustering, classification, regression, and statistical modeling to prepare intelligent applications. Also, it is a scalable machine-learning library.

Apache Mahout is distributed under a commercially friendly Apache software license. The goal of Apache Mahout is to build a vibrant, responsive, and diverse community to facilitate discussions not only on the project itself but also on potential use cases.

The following are some companies that are using Mahout:

- **Amazon**: This is a shopping portal for providing personalization recommendation
- **AOL**: This is a shopping portal for shopping recommendations
- **Drupal**: This is a PHP content management system using Mahout for providing open source content-based recommendation
- **iOffer**: This is a shopping portal, which uses Mahout's Frequent Pattern Set Mining and collaborative filtering to recommend items to users
- **LucidWorks Big Data**: This is a popular analytics firm, which uses Mahout for clustering, duplicate document detection, phase extraction, and classification
- **Radoop**: This provides a drag-and-drop interface for Big Data analytics, including Mahout clustering and classification algorithms
- **Twitter**: This is a social networking site, which uses Mahout's Latent Dirichlet Allocation (LDA) implementation for user interest modeling and maintains a fork of Mahout on GitHub.
- **Yahoo!**: This is the world's most popular web service provider, which uses Mahout's Frequent Pattern Set Mining for Yahoo! Mail

The reference links on the Hadoop ecosystem can be found at http://www.revelytix.com/?q=content/hadoop-ecosystem.

For More Information:
Apache HBase is a distributed Big Data store for Hadoop. This allows random, real-time read/write access to Big Data. This is designed as a column-oriented data storage model innovated after inspired by Google BigTable.

The following are the companies using HBase:

- **Yahoo!**: This is the world’s popular web service provider for near duplicate document detection
- **Twitter**: This is a social networking site for version control storage and retrieval
- **Mahalo**: This is a knowledge sharing service for similar content recommendation
- **NING**: This is a social network service provider for real-time analytics and reporting
- **StumbleUpon**: This is a universal personalized recommender system, real-time data storage, and data analytics platform
- **Veoh**: This is an online multimedia content sharing platform for user profiling system

For Google Big Data, distributed storage system for structured data, refer the link http://research.google.com/archive/bigtable.html.

Hive is a Hadoop-based data warehousing like framework developed by Facebook. It allows users to fire queries in SQL-like languages, such as HiveQL, which are highly abstracted to Hadoop MapReduce. This allows SQL programmers with no MapReduce experience to use the warehouse and makes it easier to integrate with business intelligence and visualization tools for real-time query processing.

Pig is a Hadoop-based open source platform for analyzing the large scale datasets via its own SQL-like language: Pig Latin. This provides a simple operation and programming interface for massive, complex data-parallelization computation. This is also easier to develop; it's more optimized and extensible. Apache Pig has been developed by Yahoo!. Currently, Yahoo! and Twitter are the primary Pig users.

For developers, the direct use of Java APIs can be tedious or error-prone, but also limits the Java programmer's use of Hadoop programming's flexibility. So, Hadoop provides two solutions that enable making Hadoop programming for dataset management and dataset analysis with MapReduce easier—these are Pig and Hive, which are always confusing.

For More Information:  
Apache Sqoop provides Hadoop data processing platform and relational databases, data warehouse, and other non-relational databases quickly transferring large amounts of data in a new way. Apache Sqoop is a mutual data tool for importing data from the relational databases to Hadoop HDFS and exporting data from HDFS to relational databases.

It works together with most modern relational databases, such as MySQL, PostgreSQL, Oracle, Microsoft SQL Server, and IBM DB2, and enterprise data warehouse. Sqoop extension API provides a way to create new connectors for the database system. Also, the Sqoop source comes up with some popular database connectors. To perform this operation, Sqoop first transforms the data into Hadoop MapReduce with some logic of database schema creation and transformation.

Apache Zookeeper is also a Hadoop subproject used for managing Hadoop, Hive, Pig, HBase, Solr, and other projects. Zookeeper is an open source distributed applications coordination service, which is designed with Fast Paxos algorithm-based synchronization and configuration and naming services such as maintenance of distributed applications. In programming, Zookeeper design is a very simple data model style, much like the system directory tree structure.

Zookeeper is divided into two parts: the server and client. For a cluster of Zookeeper servers, only one acts as a leader, which accepts and coordinates all rights. The rest of the servers are read-only copies of the master. If the leader server goes down, any other server can start serving all requests. Zookeeper clients are connected to a server on the Zookeeper service. The client sends a request, receives a response, accesses the observer events, and sends a heartbeat via a TCP connection with the server.

For a high-performance coordination service for distributed applications, Zookeeper is a centralized service for maintaining configuration information, naming, and providing distributed synchronization and group services. All these kinds of services are used in some form or another by distributed applications. Each time they are implemented, there is a lot of work that goes into fixing the bugs and race conditions that are inevitable. These services lead to management complexity when the applications are deployed.

Apache Solr is an open source enterprise search platform from the Apache license project. Apache Solr is highly scalable, supporting distributed search and index replication engine. This allows building web application with powerful text search, faceted search, real-time indexing, dynamic clustering, database integration, and rich document handling.

Apache Solr is written in Java, which runs as a standalone server to serve the search results via REST-like HTTP/XML and JSON APIs. So, this Solr server can be easily integrated with an application, which is written in other programming languages. Due to all these features, this search server is used by Netflix, AOL, CNET, and Zappos.

For More Information:
Ambari is very specific to Hortonworks. Apache Ambari is a web-based tool that supports Apache Hadoop cluster supply, management, and monitoring. Ambari handles most of the Hadoop components, including HDFS, MapReduce, Hive, Pig, HBase, Zookeeper, Sqoop, and HCatalog as centralized management.

In addition, Ambari is able to install security based on the Kerberos authentication protocol over the Hadoop cluster. Also, it provides role-based user authentication, authorization, and auditing functions for users to manage integrated LDAP and Active Directory.

Summary

In this chapter, we learned what is R, Hadoop, and their features, and how to install them before going on to analyzing the datasets with R and Hadoop. In the next chapter, we are going to learn what MapReduce is and how to develop MapReduce programs with Apache Hadoop.

For More Information:
Where to buy this book


Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.