1. Setting up Hadoop AMI AWS
2. Steps to set up the AMI on AWS:
   1. Log In to the AWS Console
   2. Click on Launch Instance
   3. Choose the AMI (Ubuntu Server – 64bit)
   4. Set the instance type to Small
   5. Set the number of instances to “4”
   6. Add Storage. Min is 8Gib.
   7. Set the Name of the AWS AMI to “HadoopNameNode”
   8. Create a new Security Group and give it a name
   9. Review all the details
   10. Click on Launch the instance
   11. Create a key and a .pem certificate for secure log in.
   12. Launch the instances
3. Connect to the instances from a Mac
   1. Open Terminal
   2. Go to the directory where the .pem file is located
   3. Enter the following command :

ssh -i MY\_PEM\_FILE.pem [ubuntu@ec2-xx-xx-xx-xx.us-west-2.compute.amazonaws.com](mailto:ubuntu@ec2-xx-xx-xx-xx.us-west-2.compute.amazonaws.com)

* 1. Ubuntu is the username, MY\_PEM\_FILE is the pem file downloaded while setting up the AWS instances.

1. Transfer files to the AWS Instance
   1. Open FileZilla
   2. Open Site Manager
   3. Make a new site
   4. Add the public IP name in the Host.
   5. Add the Port No specified while creating the instances
   6. Set the Log on type as Key File.
   7. Add the pem File to the file’s section
   8. Put the default user name for AWS Ubuntu server as Ubuntu
   9. Click on Connect
2. Setting up Hadoop in Ubuntu
   1. Modify the hostname as the Public IP of AWS by the help of command:
      1. Sudo hostname ec2-xx-xx-xx-xx.us-west-2.compute.amazonaws.com
      2. Change in the /etc/hosts file by modifying the localhost to a PrivateIP<space>PublicIP. Save the changes.
   2. Install Java
      1. Update the packages and dependencies
      2. Update and install JDK
   3. Install Hadoop
      1. Install wget services
      2. Download Hadoop latest release from Apache with the help of wget
      3. Untar the Hadoop download by :
         1. tar -xzvf hadoop-2.7.3.tar.gz
         2. Rename hadoop-2.7.3 as Hadoop
      4. Set up environment variables
         1. Go to the home directory
         2. Open the ~/.bashrc file
         3. Add variables for JAVA\_HOME, HADOOP\_HOME, HADOOP\_CONF directories
         4. Save and exit
      5. Configure SSH for password-less connection to the data node. (Only for the Name Node)
         1. Open an SSH Agent by command: eval `ssh-agent`
         2. Add the PEM file downloaded while configuring the AWS to the server with the help of FileZilla
         3. Add the PEM file to the SSH-AGENT by: ssh-add PEM\_FILE.pem
      6. Set up Hadoop Clusters
         1. Modify hadoop-env.sh and set the JAVA\_HOME directory
         2. Add the default File system Node name and Temp directory location to the core-site.xml
         3. Add DFS permissions and Replication Number (2 in our case) to the hdfs-site.xml
         4. Set up Job tracker on the Name node by updating mapred-site.xml (Only for the Name Node)
         5. Securely copy all the required files to the Slaves except mapred-site.xml
      7. Configure Masters and Slaves on Name Node
         1. Add the Public IP of the Name Node in the file HADOOP\_CONF/masters.
         2. Add the Public IP of all the Data Nodes in HADOOP\_CONF/slaves.
      8. Configure Masters and Slaves on Data Nodes
         1. Add the Public IP of the Data Node itself only in the HADOOP\_CONF/slaves file
         2. Make sure to keep the masters file empty.
      9. Format the Name Node:
         1. Commane: hadoop namenode –format
      10. Start the Hadoop server
          1. Go to HADOOP\_HOME directory
          2. Open the sbin directory
          3. Start the start-dfs.sh script
      11. Check the Name Node Status:
          1. Open a web browser
          2. http://[ec2-xx-xx-xx-xx.compute-1.amazonaws.com](http://ec2-54-209-221-112.compute-1.amazonaws.com/):50070/dfshealth.jsp
          3. In command line write: hadoop dfsadmin –report

Please note: Find the Screen shots of the above process in ScreenShots.zip. The screenshots were taken while installing the Hadoop and might not be accurate to current set up.

1. Code Files

Please find HadoopWordCount.zip file that contains the Source code of the assignments.

* 1. WordCount.java:

Describes the configurations for the Map-Reduce Jobs and specifies the Input and Output of each Map-Reduce stage as well as the final Output.

* 1. WordMapper.java, WordReducer.java:

Its function is to map each State with the corresponding count for the occurrence of Case Insensitive Words: education, politics, sports and agriculture. Stage 1

* 1. StateAndWordWritable.java:

It is a class that is used to create and object that represents the State and word combination which can be used by WordMapper and WordReducer to find the corresponding count of every combination

* 1. DominantMapper.java, DominantReducer.java:

It takes the Output of the first MapReduce stage and finds the dominant word for each state Stage 2

* 1. DominantCountWrapper.java, DominantCountReducer.java:

It takes the output of the second stage of Map Reduce as the Input and for every word finds the number of states in which the particular word is dominant compared to other states. **OUTPUT for problem statement 1 Stage 3**

* 1. DominantMapper.java, RankingReducer.java:

These files take output of 1st stage as input and provide the output as a State with its Word Rank where the words that occur the most are space separated with highest at first. Stage 4

* 1. WordAndCountWritable.java:

It is a class that is used to create and object that represents the Word and Count combination which can be used by DominantMapper and RankingReducer to find the corresponding count of every combination

* 1. SameRankingMapper.java SameRankingReducer.java:

These files take the output of 4th stage as input and provide all the states that have the same word rankings as of stage 4 with comma separated values. **OUTPUT for problem statement 2 Stage 5**

1. Running the Code

Follow the steps written in README.txt file.

1. Name of the Hadoop AMI:

Kaushil's Hadoop AMI