Word-count using 11 node MPI cluster and Data Analysis of its results using ‘R’

Nimish Joshi,
Ameya Kathapurrkar,
Rushikesh Jagdale,
Tahhan Sadaf Yusuf
Department of Electrical and Computer Engineering
University of Texas at San Antonio

***Abstract* - The rapid growth of measuring paradigms has led to accumulation of “Big-Data”; leading to an urgency of data analysis. Data accumulating at what rate, its understandability and usability is the current task of Data Analysis personnel. With proper handling and appropriate analysis can lead to new results and thus these results can be used for modelling automation. This project is a prototype towards the Big Data problem. MPI based communication coupled with powerful statistical language like R has paved the way to solve complex data analytic problem. We use the same tools to demonstrate the analysis of a sample data to plot its graphs and display its meaning from supplied dataset of dictionary.**

Key-Words:- ***Big Data, R, MPI, Python, Data Analysis, count, plots, dictionary***

I. Introduction

Data now stream from daily life: from phones and credit cards and televisions and computers; from the infrastructure of cities; from sensor-equipped buildings, trains, buses, planes, bridges, and factories. The data flow so fast that the total accumulation of the past two years—a zettabyte—dwarfs the prior record of human civilization. “There is a big data revolution,” says Weather head University Professor Gary King. But it is not the quantity of data that is revolutionary. “The big data revolution is that now we can do something with the data.” The revolution lies in improved statistical and computational methods, not in the exponential growth of storage or even computational capacity, King explains. The doubling of computing power every 18 months (Moore’s Law) “is nothing compared to a big algorithm”—a set of rules that can be used to solve a problem a thousand times faster than conventional computational methods could.

The project touchbacks the Data Analytic problem and takes towards solving critical applications. The moto is to pass a data set to the MPI cluster and then count the frequency of data that occurs frequently. Scripting language like python glues the application and produces output for usage in R. The R generates statistics and produces data for simple usability and result orientation.

II. Methodology:

To serve the objective of this project one of the possible ways can be point to point communication that allows the transmittal of data between a pair of processes through sending data from one side and receiving at the other end. Message passing among 11 processes (1 master,10 slaves) is completed using comm.send(data,dest,tag) and comm.recv(source,tag) methods. The tag information allows selectivity of messages at the receiving end. The number of processes in a communicator and the calling process rank can be respectively obtained with methods Get_size() and Get_rank(). Library “sys” is imported to use it in taking input of words as arguments in master(rank=0) and send these 10 words to 10 different slaves(from rank 1 to 10). Each of the 10 slave process count number of occurrences of corresponding received word from searchtest.txt file and send the number of counts to rank 0. In R part of the project, two files are considered: one is the “output.csv” which is the output file from python that contains desired list of words and count of these and other is a dictionary file named “dictionary_new.csv”. Using read.csv() command will allow the files to be read and converted into data frame type at the same time. This data-frame format helps to handle files easily in R environment. Importing of library “gdata” and “plotrix” is necessary before performing any data analysis and plotting in R. Then data from “output.csv” is plotted and presented graphically to show the count of chosen words. In addition, logic of looking up for exact words in dictionary is implemented using is.na() and pmatch() functions. Comparing between the columns of words from two csv files using pmatch() or partial matching function can provide combination of indices of matching words. Then applying is.na() or is not available function on pmatch() will provide non-matched words. Negation of this logic provide the desired output of matching words with corresponding meaning from the dictionary. Finally the result will be shown in the console window from dictionary data frame mydata.

III. Functionality:

The words to be searched are parsed by the user. Python script will use Message Parsing Interface (MPI) for searching the words from the text file. In python collective communication is used for broadcasting text file, and point-to-point communication for parsing the words to each slave node. In this project we are using 10 nodes for searching 10 different words. Each node will run word searching logic and will output word count for respective words. Master node will collect word count from each node and generate output file which contains each word and its count. This file is parsed to R for further data analysis.

R is used to display plots for word counts in a text file and also to find meaning of respective words from Dictionary file; which is parsed to R from local directory. In graphical presentation pie chart, bar plot and 3D pie chart is been used. All the plots are generated in a PDF format and are saved to local folder. For finding meaning of a word generic function ‘is.na’ and ‘match’ are used in R.

IV. RESULTS

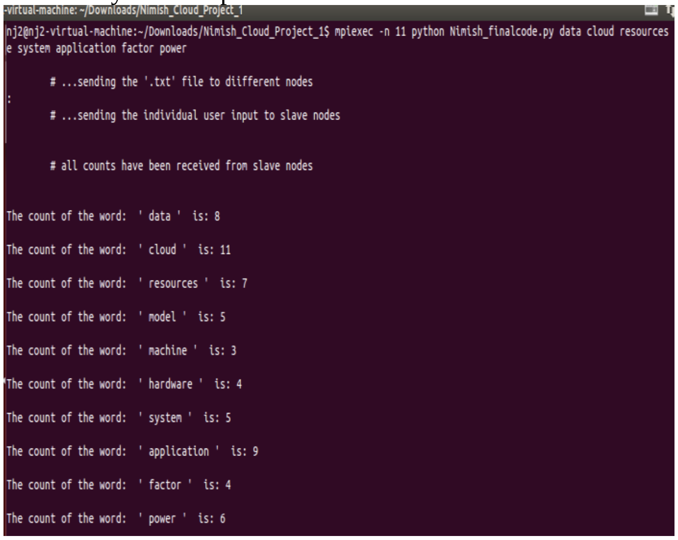
The scripts contains three essential parts, considering all required library mentioned above are preloaded into the system with proper requirements. These three parts are followed.

1. Python script
2. R-script
3. Bash script

The bash script will combine these two: python and R-script to-gather. It fulfils our motto of scripts automation to speed up the process and to reduce the manual implementation of individual script.

We will look each results one by one.

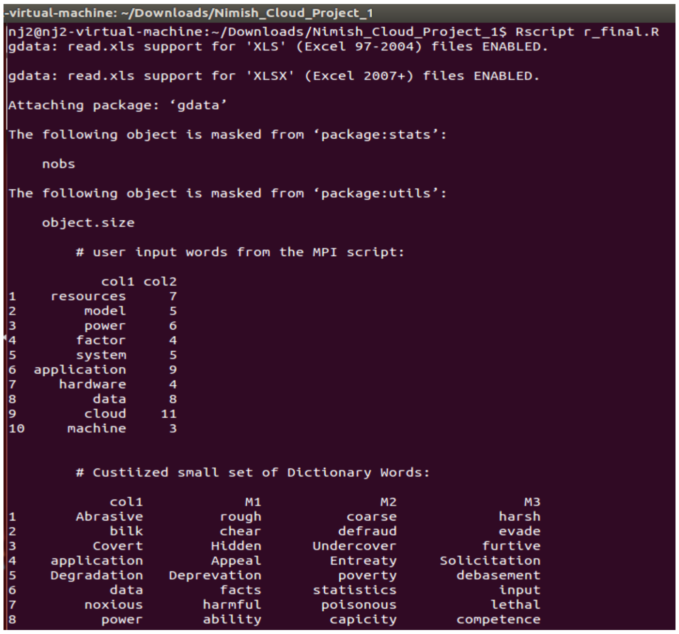
1. Python Script:



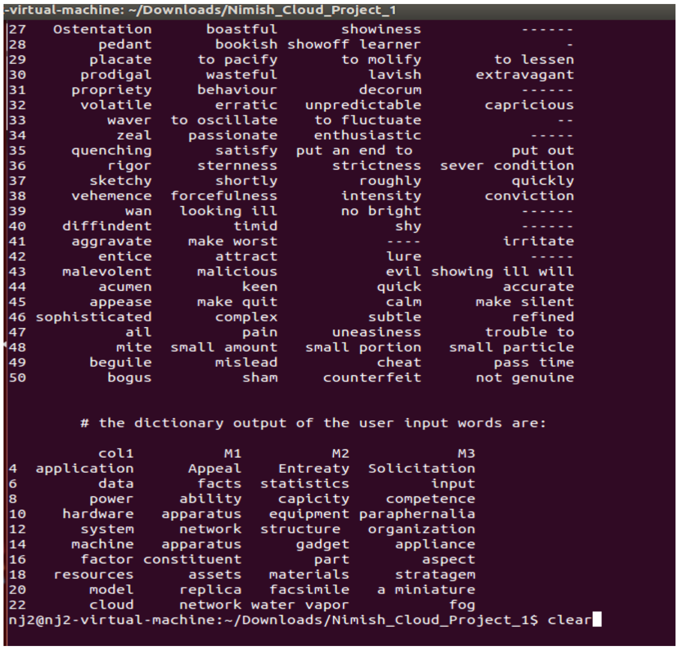
```
-virtual-machine: ~/Downloads/Nimish_Cloud_Project_1
nj2@nj2-virtual-machine:~/Downloads/Nimish_Cloud_Project_1$ mpiexec -n 11 python Nimish_finalcode.py data cloud resources
e system application factor power

        # ...sending the '.txt' file to diiferent nodes
:
        # ...sending the individual user input to slave nodes


        # all counts have been received from slave nodes


The count of the word:  ' data '  is: 8

The count of the word:  ' cloud '  is: 11

The count of the word:  ' resources '  is: 7

The count of the word:  ' model '  is: 5

The count of the word:  ' machine '  is: 3

The count of the word:  ' hardware '  is: 4

The count of the word:  ' system '  is: 5

The count of the word:  ' application '  is: 9

The count of the word:  ' factor '  is: 4

The count of the word:  ' power '  is: 6
```

In abstract manner, here master node is sharing .txt file with slave nodes which is broadcasted to them. User gives user input and distributed to all 10 different nodes: no node receive more than one word. Later is these words are being searched and will get back with its count to the master nodes. All the output from different nodes would be saved into an output file in .csv format.

2. R-script:



```
-virtual-machine: ~/Downloads/Nimish_Cloud_Project_1
nj2@nj2-virtual-machine:~/Downloads/Nimish_Cloud_Project_1$ Rscript r_final.R
gdata: read.xls support for 'XLS' (Excel 97-2004) files ENABLED.

gdata: read.xls support for 'XLSX' (Excel 2007+) files ENABLED.

Attaching package: ‘gdata’

The following object is masked from ‘package:stats’:

    nobs

The following object is masked from ‘package:utils’:

    object.size

        # user input words from the MPI script:

           col1 col2
1    resources    7
2        model    5
3        power    6
4       factor    4
5       system    5
6  application    9
7     hardware    4
8         data    8
9        cloud   11
10     machine    3


        # Custized small set of Dictionary Words:

            col1            M1             M2               M3
1      Abrasive          rough         coarse            harsh
2          bilk          chear        defraud            evade
3        Covert         Hidden     Undercover          furtive
4   application         Appeal       Entreaty      Solicitation
5   Degradation    Deprevation        poverty        debasement
6          data          facts     statistics            input
7       noxious        harmful      poisonous           lethal
8         power        ability       capicity       competence
```

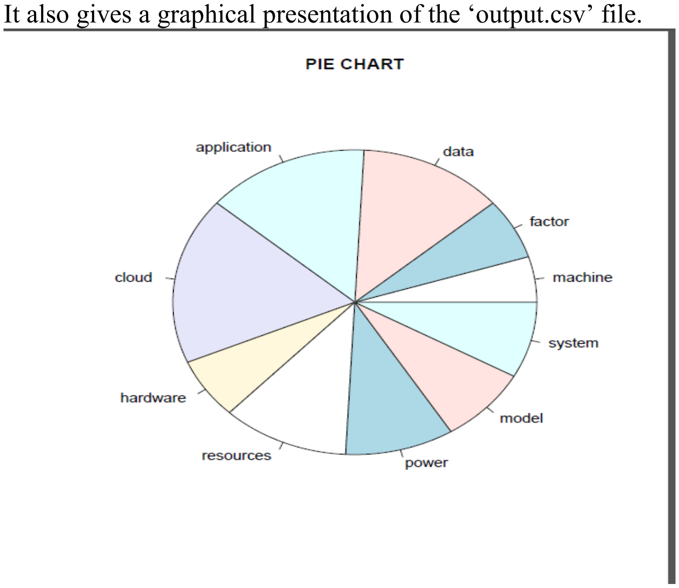


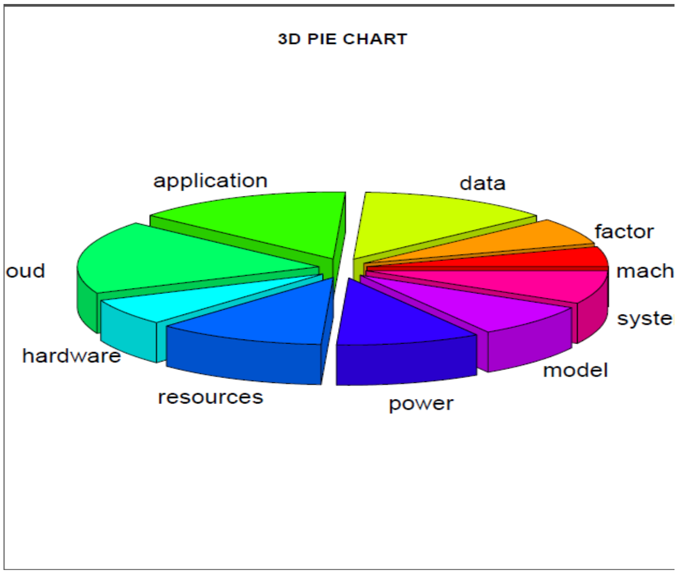
```
-virtual-machine: ~/Downloads/Nimish_Cloud_Project_1
27   Ostentation      boastful      showiness            ------
28        pedant       bookish showoff learner                -
29       placate     to pacify      to molify        to lessen
30      prodigal      wasteful         lavish      extravagant
31      propriety    behaviour        decorum           ------
32      volatile       erratic  unpredictable       capricious
33         waver to oscillate   to fluctuate               --
34          zeal   passionate   enthusiastic            -----
35     quenching      satisfy  put an end to          put out
36         rigor    sternness     strictness  sever condition
37       sketchy      shortly        roughly          quickly
38     vehemence forcefulness      intensity       conviction
39           wan  looking ill      no bright           ------
40    diffindent        timid            shy           ------
41     aggravate   make worst           ----         irritate
42        entice      attract           lure            -----
43    malevolent    malicious           evil showing ill will
44        acumen         keen          quick         accurate
45       appease    make quit           calm      make silent
46 sophisticated      complex         subtle          refined
47           ail         pain     uneasiness       trouble to
48          mite small amount   small portion   small particle
49       beguile      mislead          cheat        pass time
50         bogus         sham    counterfeit      not genuine


        # the dictionary output of the user input words are:

          col1          M1          M2            M3
4  application      Appeal    Entreaty   Solicitation
6         data       facts  statistics         input
8        power     ability    capicity    competence
10    hardware   apparatus   equipment paraphernalia
12      system     network   structure  organization
14     machine   apparatus      gadget     appliance
16      factor constituent        part        aspect
18   resources      assets   materials      stratagem
20       model     replica   facsimile   a miniature
22       cloud     network water vapor           fog
nj2@nj2-virtual-machine:~/Downloads/Nimish_Cloud_Project_1$ clear
```

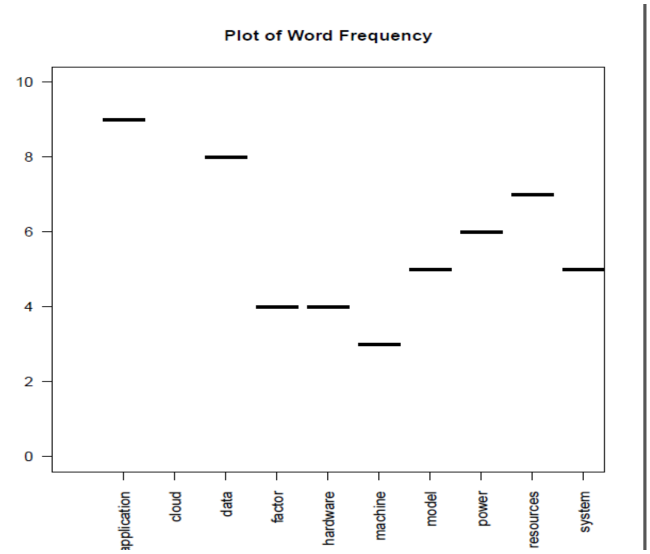
In the R-script, the output file from python code will be read and saved into the data frame which later will be compared with the dictionary database that we have made of .csv file.

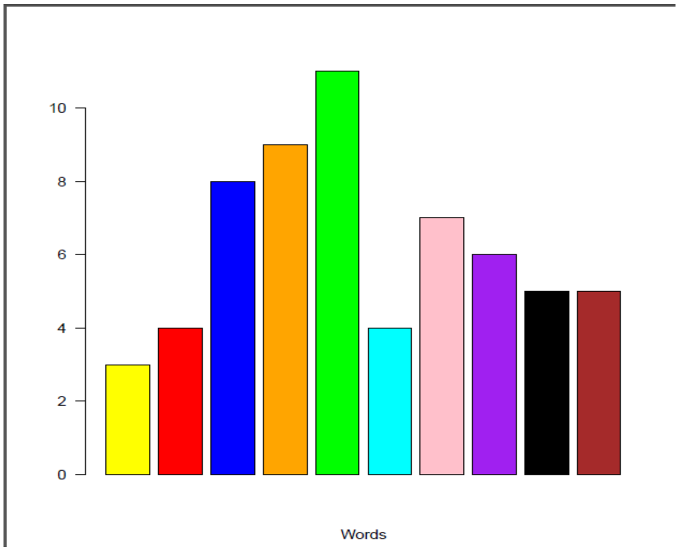
Matching of user input words will be displayed with its 3 different meanings. Furthermore, data analysis of those output.csv file will be done through graphical representation that will be saved into .pdf file.

It also gives a graphical presentation of the ‘output.csv’ file.









3. Bash-sript:

Main aim of bash script was to run both python and R program one by one and to make the process automated here.

V. CONCLUSION

By using MPI and combination of number of cluster we can optimize the resources and can get fast results. This technique further can be used into the data analysis especially where large set of database file need to be handled and analyzed. Using R into the project has added many graphical tool of the data analysis which also can be used at the large scale.

REFERENCE

"http://harvardmagazine.com/2014/03/why-big-data-is-a-big-deal"