**Introduction to Cluster Analysis**

**Setup, Compilation, Execution and Output Guidelines**

**Setup and Pre-requisites:**

Hadoop Setup and Configuration- Minimum 2.6

Environment built on Linux

Input dataset all in the Hadoop home directory i.e. /usr/local/Hadoop along with the working code provided in the tar.gz folder

Our Makefile consist of two rules, one for the program to execute on the Pseudo Distributed System and one for the EMR

For EMR, we need to have the AWS CLI installation and configuration done

All the deliverables are to be placed in the hadoop home so that we don’t face any path conflict

**Compilation and Execution:**

Pseudo Distributed System-

We are providing a jar file containing all compiled classes or one can compile the same using below commands

javac -cp /usr/local/hadoop/share/hadoop/mapreduce/lib/hadoop-annotations-2.6.3.jar:/usr/local/hadoop/share/hadoop/common/hadoop-common-2.6.3.jar:/usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-client-core-2.6.3.jar:/usr/local/hadoop/opencsv-2.2.jar  HW02.java

jar cvf HW02.jar HW02\*.class

run the rule – make pseudo // This will start the Hadoop system, run the job and provide the output at <Hadoop-home>/output

EMR-

For the cluster setup and code to run on EMR, follow the rule

make emr

This will create a cluster, run the jar file at the mentioned location and terminates the cluster on the execution of the given job.

Results ->

We could successfully establish the HDFS install/Configure and setup for MR. This code involves 25 mappers for 25 csv files as input dataset and 4 Reducer with generated 4 output files

part-r-00000

part-r-00001

part-r-00002

part-r-00003 txt files

These files contain the output from the 4 reducers.

Parallelism is achieved via these mappers as well as via the 4 reducers. The result from these output is then merged to a common file on the local Hadoop-home directory by running the below script

bin/hadoop dfs –getmerge /user/HW02/output/part\* /usr/local/hadoop/final\_output.txt

Conclusion – For this large dataset, the time taken for the entire processing is observed as decreasing as we increase the number of reducers for 25 set of mappers.

Efficient parallelism is achieved

**References:**

[https://docs.google.com/document/d/18HkaZ0tSAH8xjDOZjPULBfAJw5H\_vj33QSLemseRM2o/edit?pref=2&pli=1#](https://docs.google.com/document/d/18HkaZ0tSAH8xjDOZjPULBfAJw5H_vj33QSLemseRM2o/edit?pref=2&pli=1)

http://stackoverflow.com/

<https://developer.yahoo.com/hadoop/tutorial/module2.html>

<https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SingleCluster.html>

---------------------------------------END--------------------------------------------------