

Zafar Mahmood
Distributed Lab
Solution 5
17 Mai 2017

Hadoop 2.7.2

Subversion <https://git-wip-us.apache.org/repos/asf/hadoop.git> -r

b165c4fe8a74265c792ce23f546c64604acf0e41

Compiled by jenkins on 2016-01-26T00:08Z

Compiled with protoc 2.5.0

From source with checksum d0fda26633fa762bff87ec759ebe689c

This command was run using /home/zfar/hadoop-2.7.2/share/hadoop/common/hadoop-common-2.7.2.jar

Hadoop Configurations : (Browse the file system)

Hadoop Overview Datanodes Snapshot Startup Progress Utilities -

Browse Directory

Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	zfar	supergroup	0 B	5/17/2017, 1:06:06 AM	0	0 B	hadoopdemo
drwxr-xr-x	zfar	supergroup	0 B	5/18/2017, 11:41:13 AM	0	0 B	user

Hadoop, 2015.

Exercise 2 : Analysis Of Airport

- **Data Filtration Applied !**
 - Geography – All
 - Year – 2017
 - Filter Period – January
 - Time Period – Flight Date
 - Origin – Origin
 - Destination – Destination
 - Departure Performance - DepTime, DepDelay
 - Arrival Performance: ArrTime, ArrDelay
- **Computing the maximum, minimum, and average departure delay for each airport**
 - Hadoop Steps
 - Upload the data file to Data Node
 - `hdfs dfs -put hadoop-2.7.2/hadoop/movieLen/output.dat /user/zfar/airLine/Input`

Browse Directory

/user/zfar/airlineInput							Go!
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	zfar	supergroup	45.2 MB	5/19/2017, 12:32:28 AM	1	128 MB	sample_data.csv

Hadoop, 2015.

- Algorithm steps Steps Followed
 - **Mapper file**
 - Receives the input **sys**
 - Now strip and split the data “ :: ”
 - Now take the column that needed
 - Delay and Airport Names
 - **Pass it reducer function**
 - Now it will start checking the airports names , present and old **as it come sorted**
 - Now a list has been make for each airport , which will store its min , max and average delay time, in the end it get displayed in hadoop -output directory

```
ABE :: 9 9 9.0
ABQ :: 26 41 33.5
ABY :: -9 48 19.5
ACT :: -10 -7 -8.5
AEX :: -2 354 143.25
AGS :: -6 88 41.0
ALB :: -4 -3 -3.5
ANC :: -10 -6 -8.3333333333333
ATL :: -7 400 31.6986969471
ATW :: -7 -7 -7.0
AUS :: -9 63 6.20574434807
AVL :: -7 6 0.25
BDL :: -10 27 8.28193456615
BHM :: -8 9 0.5
BLI :: 1 245 92.3333333333333
BMI :: -11 -10 -10.5
BNA :: -17 1447 110.95192718
BOI :: 0 0 0.0
BOS :: -11 160 8.43913718607
BPT :: -6 -6 -6.0
BQK :: -1 -1 -1.0
```

- using hadoop run command

```
hadoop jar hadoop-2.7.2/share/hadoop/tools/lib/hadoop-streaming-2.7.2.jar -file hadoop-2.7.2/hadoop/airline/mapper.py
-mapper "python2.7 mapper.py" -file hadoop-2.7.2/hadoop/airline/reducer.py -reducer "python2.7 reducer.py" -input
/user/zfar/airline/input -output /user/zfar/airline/output
```

Browse Directory

/user/zfar/airlineOutput							Go!
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	zfar	supergroup	0 B	5/19/2017, 12:36:49 AM	1	128 MB	_SUCCESS
-rw-r--r--	zfar	supergroup	199 B	5/19/2017, 12:36:49 AM	1	128 MB	part-00000

Hadoop, 2015.

- **Computing a ranking list that contains top 10 airports by their average Arrival delay**
 - Algorithm is used same as above but in the end we have names of all airports but here its evaluated using their arrival delay
 - After when the its computer , sort arrival delay and display the top 10 airports
 - file – mapper_arrival.py , reducer_arrival.py

```
[ 'FNT', 'ERI', 'PIB', 'BQK', 'DHN', 'DAY', 'ABE', 'EWN', 'FSM', 'BOI' ]
[ 606.0, 640.0, 715.0, 722.0, 732.0, 749.0, 750.0, 825.0, 840.0, 851.0 ]
```

Exercise : Analysis of Movie dataset using Map and Reduce

Preprocessed : Merged two files *

Part 1 : Highest Average Rating Movie Title

Algorithm

Mapper function : passed key, value [Movie title , rating]

Reducer function : Calculate the average of each movie and result back

Files Attached (mapper_title.py , reducer_title.py & all other files with title are outputs with different mapper and reducers)

	Mapper_1	Mapper_3
Reducer_1	172.080148935318	161.023507
Reducer_3	83.188 , 125.1353 , 164.94	78.066 , 116.282 , 154.0162

Part 2 : Find the user who has assign lowest average rating among all the users who rated more than 40 times

Algorithm :

As this is the simple algorithm

- Take a userID that is coming to reducers from the mapper ,
- Now count the number of items (rated items) that it received from mapper store it in an array
- Now when the new user comes with different ID , we will check the length of stored ratings from array , If its > 40 then take mean and store it in final array
- For finalizing we just take the minimum rating corresponding to index for User ID

Files Attached (40_User_mapper.py , 40 User_reducer.py and all other files with name 40 user are output files with different mappers and reducers)

With Short Sample Data

```
Total Number of Users who rated more than 40 times 9
User with lowest ID : 13 with rating 3.27586206897
```

Part 3 : Find the highest average rated movie genre

Algorithm:

The algorithm is same as the old one is to keep track of genre (As this is used as a key in this question)

but before it has been merged (Movie and Ranking) based on their ID's

Tested On small Data

```
Best Ratio Genre  Drama|Mystery
rating 3.91463414634
total program time : 0.9292268753051758
```

Files attached (mapper_t_3.py , reducer_t_3.py and all files with t_3 are output files)

	Reducer _ 1	Reducer _ 3
Mapper _ 1	118.14	38.103, 115.98 , 137.13
Mapper _ 3	121.726	67.5201 , 81.92 , 133.260