NBA Shot Logs

For each pair of the players (A, B), we define the fear score of A when facing B as the hit rate, such that B is the closest defender when A is shooting. Based on the fear sore, for each player, please find out who is his ”most unwanted defender”.

Mapper 1: Filtering the data with respect to defender,player and shot and assigning a value 1 if shot is made or 0 if missed.

```
import sys
mapper1_output=[]
for line in sys.stdin:
  line=line.strip(',')
  name= line.split(",")[-2]
  shot= '1' if line.split(",")[14] == 'made' else '0'
  defen_las=line.split(",")[15].strip("")
  defen_fir=line.split(",")[16].strip("")
  defen_ful=defen_fir+" "+defen_las
  defen_ful.strip("")
  print(name+','+defen_ful+'\t'+ shot)
```

Reducer 1: Splitting the input and saving it in a dictionary where the key value is player and defender. Counter is set to calculate the total hits and save it as a string alone separated from the count of shots that have been made by “\t”.

```
import sys
from operator import itemgetter
dict_score_count = {}
reducer_1_ouput = []
for line in sys.stdin:
  record = line.split('\t')
  data,count = record[0], record[1]
  try:
     count = int(count)
```

```
dict_score_count[data] = [dict_score_count.get(data, [0,0])[0] + count,
dict_score_count.get(data, [0,0])[1]+1]
   except ValueError:
       pass
for key, value in dict_score_count.items():
 print(key+'\t'+str(value))
```

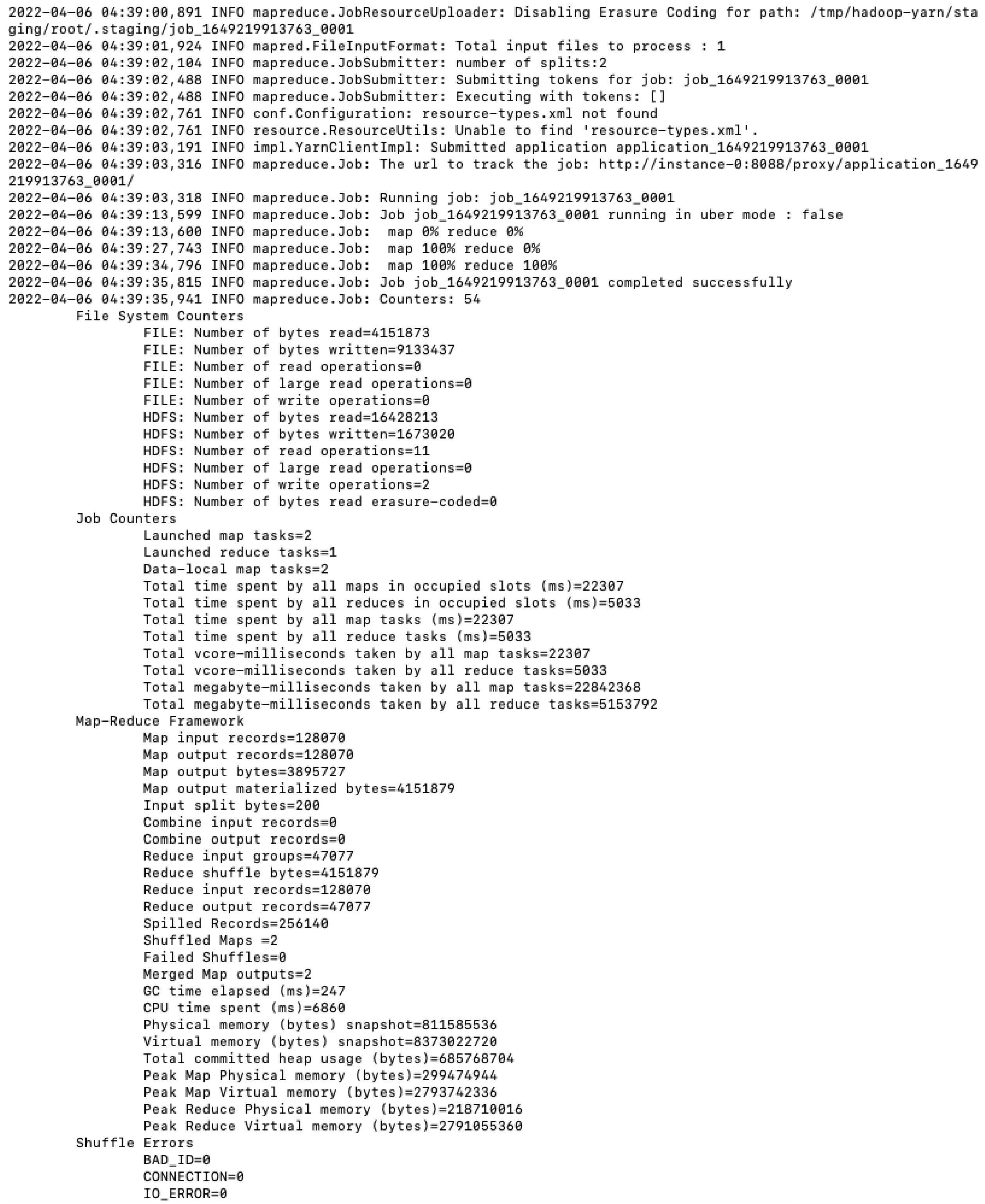
Mapper 2: Calculating fear score for each player. The fear score is the sum of made shots divided by the number of attempts the player made (hit rate). So if we get a low hit rate, that means we are getting a high fear score (inverse relationship).

```
import sys
for line in sys.stdin:
 players_pairs, result_shots = line.split('\t')
 attacker, defender = players_pairs.strip().split(',')
 attacker = attacker.strip(); defender = defender.strip()
 try:
   result_shots = eval(result_shots)
   if result_shots[1] == 1 and result_shots[0] == 0:
     continue
   else:
     result_shots = result_shots[0]/result_shots[1]
 except:
   pass
 print(attacker + '\t' + defender + ',' + str(result_shots))
```

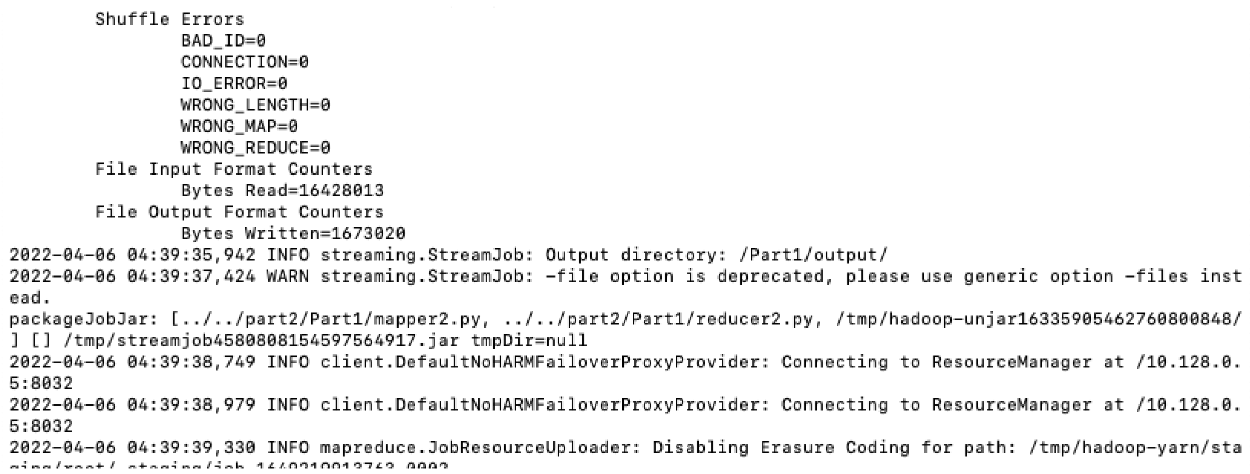
Reducer 2: Sorting values using attacker/player as key. For each attacker we sort the values in an ascending order so we get one defender with the highest **fear score**.

```
import sys

reducer_2_output = {}
for entry in sys.stdin:
   attacker, defender_score = entry.split('\t')
   defender, score = defender_score.split(',')
   score = float(score)

   if attacker not in reducer_2_output:
     reducer_2_output[attacker] = [(defender, score)]
   else:
     reducer_2_output[attacker].append((defender, score))

for player, defender_score in reducer_2_output.items():
 reducer_2_output[player] = sorted(reducer_2_output[player], key=lambda x: x[1])[0]
 print(player+'\t'+reducer_2_output[player][0])
```

Mapper 1 and Reducer 1



```
2022-04-06 04:39:00,891 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/sta
ging/root/.staging/job_1649219913763_0001
2022-04-06 04:39:01,924 INFO mapred.FileInputFormat: Total input files to process : 1
2022-04-06 04:39:02,104 INFO mapreduce.JobSubmitter: number of splits:2
2022-04-06 04:39:02,488 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1649219913763_0001
2022-04-06 04:39:02,488 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-04-06 04:39:02,761 INFO conf.Configuration: resource-types.xml not found
2022-04-06 04:39:02,761 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-04-06 04:39:03,191 INFO impl.YarnClientImpl: Submitted application application_1649219913763_0001
2022-04-06 04:39:03,316 INFO mapreduce.Job: The url to track the job: http://instance-0:8088/proxy/application_1649
219913763_0001/
2022-04-06 04:39:03,318 INFO mapreduce.Job: Running job: job_1649219913763_0001
2022-04-06 04:39:13,599 INFO mapreduce.Job: Job job_1649219913763_0001 running in uber mode : false
2022-04-06 04:39:13,600 INFO mapreduce.Job:  map 0% reduce 0%
2022-04-06 04:39:27,743 INFO mapreduce.Job:  map 100% reduce 0%
2022-04-06 04:39:34,796 INFO mapreduce.Job:  map 100% reduce 100%
2022-04-06 04:39:35,815 INFO mapreduce.Job: Job job_1649219913763_0001 completed successfully
2022-04-06 04:39:35,941 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=4151873
                FILE: Number of bytes written=9133437
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=16428213
                HDFS: Number of bytes written=1673020
                HDFS: Number of read operations=11
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=22307
                Total time spent by all reduces in occupied slots (ms)=5033
                Total time spent by all map tasks (ms)=22307
                Total time spent by all reduce tasks (ms)=5033
                Total vcore-milliseconds taken by all map tasks=22307
                Total vcore-milliseconds taken by all reduce tasks=5033
                Total megabyte-milliseconds taken by all map tasks=22842368
                Total megabyte-milliseconds taken by all reduce tasks=5153792
        Map-Reduce Framework
                Map input records=128070
                Map output records=128070
                Map output bytes=3895727
                Map output materialized bytes=4151879
                Input split bytes=200
                Combine input records=0
                Combine output records=0
                Reduce input groups=47077
                Reduce shuffle bytes=4151879
                Reduce input records=128070
                Reduce output records=47077
                Spilled Records=256140
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=247
                CPU time spent (ms)=6860
                Physical memory (bytes) snapshot=811585536
                Virtual memory (bytes) snapshot=8373022720
                Total committed heap usage (bytes)=685768704
                Peak Map Physical memory (bytes)=299474944
                Peak Map Virtual memory (bytes)=2793742336
                Peak Reduce Physical memory (bytes)=218710016
                Peak Reduce Virtual memory (bytes)=2791055360
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
```

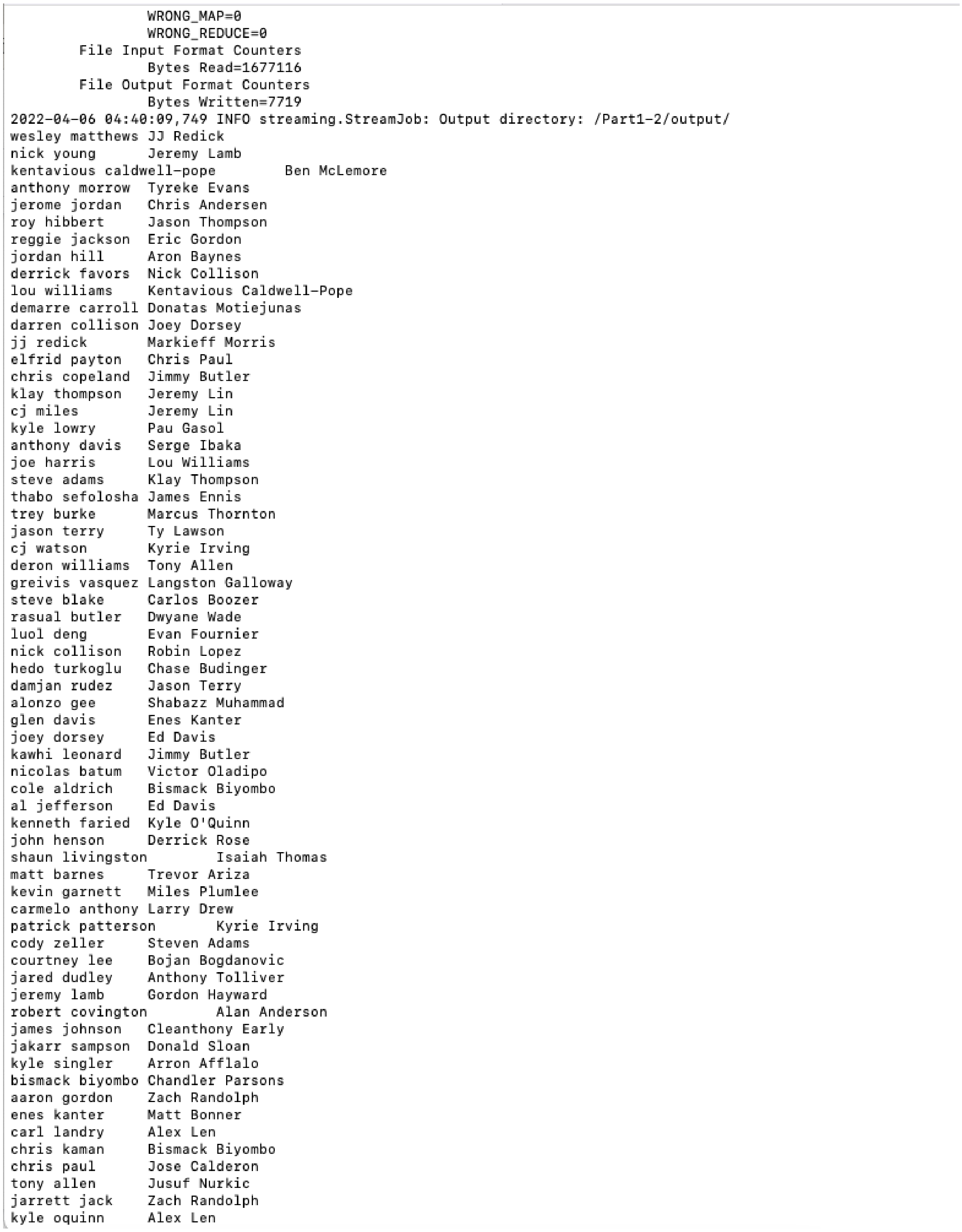


```
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=16428013
        File Output Format Counters
                Bytes Written=1673020
2022-04-06 04:39:35,942 INFO streaming.StreamJob: Output directory: /Part1/output/
2022-04-06 04:39:37,424 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files inst
ead.
packageJobJar: [../../part2/Part1/mapper2.py, ../../part2/Part1/reducer2.py, /tmp/hadoop-unjar1633590546276080848/
] [] /tmp/streamjob4580808154597564917.jar tmpDir=null
2022-04-06 04:39:38,749 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /10.128.0.
5:8032
2022-04-06 04:39:38,979 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /10.128.0.
5:8032
2022-04-06 04:39:39,330 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/sta
ging/root/.staging/job_1649219913763_0003
```

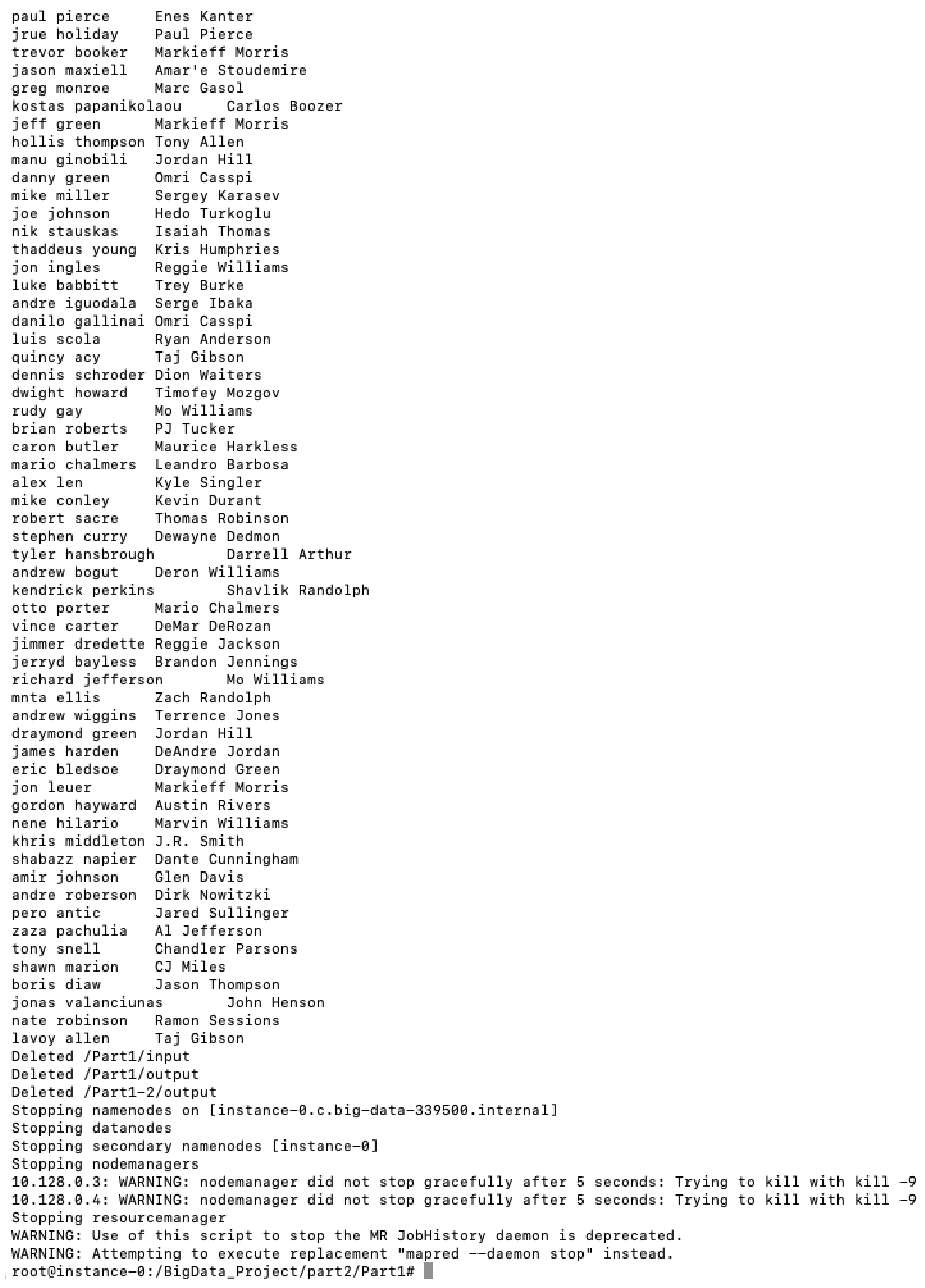
\

Mapper 2 and Reducer 2

```
2022-04-02 17:16:34,198 INFO mapreduce.Job: Job job_1648919715157_0002 running in uber mode : false
2022-04-02 17:16:34,199 INFO mapreduce.Job:  map 0% reduce 0%
2022-04-02 17:16:45,328 INFO mapreduce.Job:  map 50% reduce 0%
2022-04-02 17:16:46,336 INFO mapreduce.Job:  map 100% reduce 0%
2022-04-02 17:16:53,392 INFO mapreduce.Job:  map 100% reduce 100%
2022-04-02 17:16:53,404 INFO mapreduce.Job: Job job_1648919715157_0002 completed successfully
2022-04-02 17:16:53,508 INFO mapreduce.Job: Counters: 54
	File System Counters
		FILE: Number of bytes read=1141360
		FILE: Number of bytes written=3112417
		FILE: Number of read operations=0
		FILE: Number of large read operations=0
		FILE: Number of write operations=0
		HDFS: Number of bytes read=1677312
		HDFS: Number of bytes written=11374
		HDFS: Number of read operations=11
		HDFS: Number of large read operations=0
		HDFS: Number of write operations=2
		HDFS: Number of bytes read erasure-coded=0
	Job Counters
		Launched map tasks=2
		Launched reduce tasks=1
		Data-local map tasks=2
		Total time spent by all maps in occupied slots (ms)=17895
		Total time spent by all reduces in occupied slots (ms)=4566
		Total time spent by all map tasks (ms)=17895
		Total time spent by all reduce tasks (ms)=4566
		Total vcore-milliseconds taken by all map tasks=17895
		Total vcore-milliseconds taken by all reduce tasks=4566
		Total megabyte-milliseconds taken by all map tasks=18324480
		Total megabyte-milliseconds taken by all reduce tasks=4675584
	Map-Reduce Framework
		Map input records=47077
		Map output records=36227
		Map output bytes=1068900
		Map output materialized bytes=1141366
		Input split bytes=196
		Combine input records=0
		Combine output records=0
		Reduce input groups=281
		Reduce shuffle bytes=1141366
		Reduce input records=36227
		Reduce output records=1
		Spilled Records=72454
		Shuffled Maps =2
		Failed Shuffles=0
		Merged Map outputs=2
		GC time elapsed (ms)=222
		CPU time spent (ms)=4960
		Physical memory (bytes) snapshot=790540288
		Virtual memory (bytes) snapshot=8378494976
		Total committed heap usage (bytes)=622854144
```



```
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=1677116
        File Output Format Counters
                Bytes Written=7719
2022-04-06 04:40:09,749 INFO streaming.StreamJob: Output directory: /Part1-2/output/
wesley matthews JJ Redick
nick young      Jeremy Lamb
kentavious caldwell-pope        Ben McLemore
anthony morrow  Tyreke Evans
jerome jordan   Chris Andersen
roy hibbert     Jason Thompson
reggie jackson  Eric Gordon
jordan hill     Aron Baynes
derrick favors  Nick Collison
lou williams    Kentavious Caldwell-Pope
demarre carroll Donatas Motiejunas
darren collison Joey Dorsey
jj redick       Markieff Morris
elfrid payton   Chris Paul
chris copeland  Jimmy Butler
klay thompson   Jeremy Lin
cj miles        Jeremy Lin
kyle lowry      Pau Gasol
anthony davis   Serge Ibaka
joe harris      Lou Williams
steve adams     Klay Thompson
thabo sefolosha James Ennis
trey burke      Marcus Thornton
jason terry     Ty Lawson
cj watson       Kyrie Irving
deron williams  Tony Allen
greivis vasquez Langston Galloway
steve blake     Carlos Boozer
rasual butler   Dwyane Wade
luol deng       Evan Fournier
nick collison   Robin Lopez
hedo turkoglu   Chase Budinger
damjan rudez    Jason Terry
alonzo gee      Shabazz Muhammad
glen davis      Enes Kanter
joey dorsey     Ed Davis
kawhi leonard   Jimmy Butler
nicolas batum   Victor Oladipo
cole aldrich    Bismack Biyombo
al jefferson    Ed Davis
kenneth faried  Kyle O'Quinn
john henson     Derrick Rose
shaun livingston        Isaiah Thomas
matt barnes     Trevor Ariza
kevin garnett   Miles Plumlee
carmelo anthony Larry Drew
patrick patterson       Kyrie Irving
cody zeller     Steven Adams
courtney lee    Bojan Bogdanovic
jared dudley    Anthony Tolliver
jeremy lamb     Gordon Hayward
robert covington        Alan Anderson
james johnson   Cleanthony Early
jakarr sampson  Donald Sloan
kyle singler    Arron Afflalo
bismack biyombo Chandler Parsons
aaron gordon    Zach Randolph
enes kanter     Matt Bonner
carl landry     Alex Len
chris kaman     Bismack Biyombo
chris paul      Jose Calderon
tony allen      Jusuf Nurkic
jarrett jack    Zach Randolph
kyle oquinn     Alex Len
```



```
paul pierce     Enes Kanter
jrue holiday    Paul Pierce
trevor booker   Markieff Morris
jason maxiell   Amar'e Stoudemire
greg monroe     Marc Gasol
kostas papanikolaou     Carlos Boozer
jeff green      Markieff Morris
hollis thompson Tony Allen
manu ginobili   Jordan Hill
danny green     Omri Casspi
mike miller     Sergey Karasev
joe johnson     Hedo Turkoglu
nik stauskas    Isaiah Thomas
thaddeus young  Kris Humphries
jon ingles      Reggie Williams
luke babbitt    Trey Burke
andre iguodala  Serge Ibaka
danilo gallinai Omri Casspi
luis scola      Ryan Anderson
quincy acy      Taj Gibson
dennis schroder Dion Waiters
dwight howard   Timofey Mozgov
rudy gay        Mo Williams
brian roberts   PJ Tucker
caron butler    Maurice Harkless
mario chalmers  Leandro Barbosa
alex len        Kyle Singler
mike conley     Kevin Durant
robert sacre    Thomas Robinson
stephen curry   Dewayne Dedmon
tyler hansbrough        Darrell Arthur
andrew bogut    Deron Williams
kendrick perkins        Shavlik Randolph
otto porter     Mario Chalmers
vince carter    DeMar DeRozan
jimmer dredette Reggie Jackson
jerryd bayless  Brandon Jennings
richard jefferson       Mo Williams
mnta ellis      Zach Randolph
andrew wiggins  Terrence Jones
draymond green  Jordan Hill
james harden    DeAndre Jordan
eric bledsoe    Draymond Green
jon leuer       Markieff Morris
gordon hayward  Austin Rivers
nene hilario    Marvin Williams
khris middleton J.R. Smith
shabazz napier  Dante Cunningham
amir johnson    Glen Davis
andre roberson  Dirk Nowitzki
pero antic      Jared Sullinger
zaza pachulia   Al Jefferson
tony snell      Chandler Parsons
shawn marion    CJ Miles
boris diaw      Jason Thompson
jonas valanciunas       John Henson
nate robinson   Ramon Sessions
lavoy allen     Taj Gibson
Deleted /Part1/input
Deleted /Part1/output
Deleted /Part1-2/output
Stopping namenodes on [instance-0.c.big-data-339500.internal]
Stopping datanodes
Stopping secondary namenodes [instance-0]
Stopping nodemanagers
10.128.0.3: WARNING: nodemanager did not stop gracefully after 5 seconds: Trying to kill with kill -9
10.128.0.4: WARNING: nodemanager did not stop gracefully after 5 seconds: Trying to kill with kill -9
Stopping resourcemanager
WARNING: Use of this script to stop the MR JobHistory daemon is deprecated.
WARNING: Attempting to execute replacement "mapred --daemon stop" instead.
root@instance-0:/BigData_Project/part2/Part1# 
```

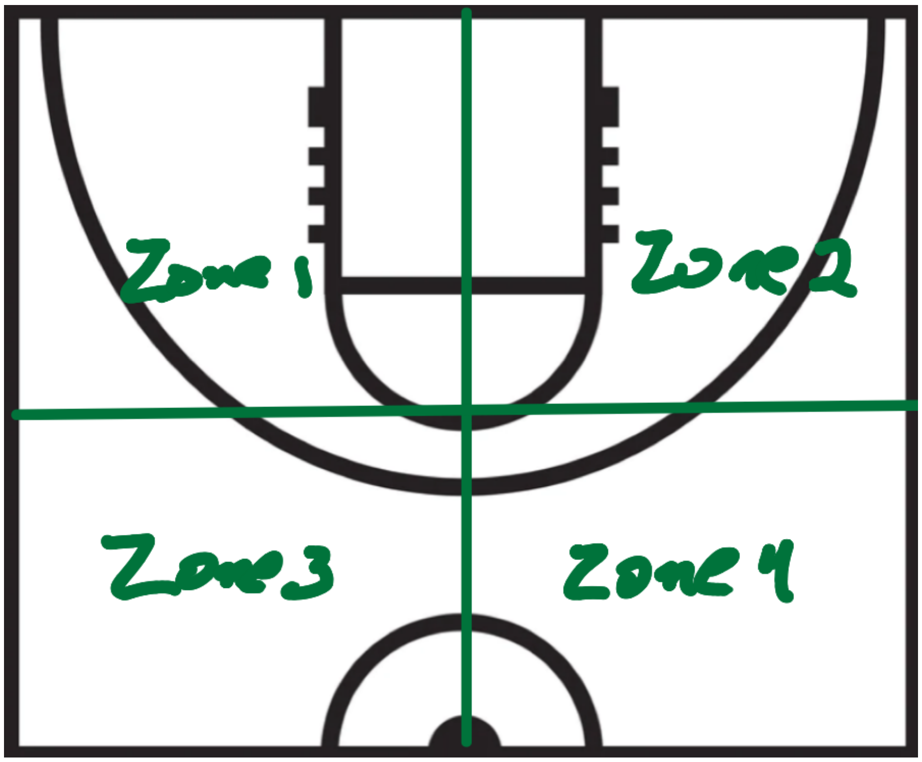
Part 2: For each player, we define the comfortable zone of shooting is a matrix of,

{SHOT DIST, CLOSE DEF DIST, SHOT CLOCK}

Please develop a MapReduce-based algorithm to classify each player's records into 4 comfortable zones. Considering the hit rate, which zone is the best for James Harden, Chris Paul, Stephen Curry and Lebron James.

Mapper 1: Creating a random cluster and then calculating the euclidean distance of each data point from each of the clusters and assigning it the key of the cluster with minimum distance. By using K -means clustering, we are finding 4 clusters for the matrix {SHOT DIST, CLOSE DEF DIST, SHOT CLOCK}.

For the clustering process we made use of SHOT_DIST < 29 and CLOSE_DEF_DIST < 11 and SHOT_CLOCK < 26, being a vectorial space without outliers and related to a space that represents the attacking zone of the basketball court.



```
import sys
zones_mapper = sys.argv[1]
zones_mapper = [eval(dp) for dp in zones_mapper.split('Z')[1].strip('_').split('_')]
zones_mapper = {1:[zones_mapper[0],zones_mapper[1],zones_mapper[2]],
               2:[zones_mapper[3],zones_mapper[4],zones_mapper[5]],
               3:[zones_mapper[6],zones_mapper[7],zones_mapper[8]],
               4:[zones_mapper[9],zones_mapper[10],zones_mapper[11]]}
def euclidean_distance(A, B):
 return sum((a-b)**2 for a, b in zip(A[:], B[:])) ** (1/2)
mapper_1_output = {1:[0,[0,0,0]],
                 2:[0,[0,0,0]],
                 3:[0,[0,0,0]],
                 4:[0,[0,0,0]]}
for line in sys.stdin:
 line = line.strip(',').split(',')
 line_len = len(line)
 if line_len == 23:
   try:
     SHOT_DIST = float(line[12].strip('"'))
     CLOSE_DEF_DIST= float(line[-5].strip('"'))
     SHOT_CLOCK = float(line[9].strip('"'))
     if SHOT_DIST < 29 and CLOSE_DEF_DIST < 11 and SHOT_CLOCK < 26: # removing
outliers
       data = [SHOT_DIST, CLOSE_DEF_DIST, SHOT_CLOCK]
       data_centroids_distances = {1: euclidean_distance(data, zones_mapper[1]),
                             2: euclidean_distance(data, zones_mapper[2]),
                             3: euclidean_distance(data, zones_mapper[3]),
                             4: euclidean_distance(data, zones_mapper[4])}
```

```
    data_cluster_key = min(data_centroids_distances, key = data_centroids_distances.get)
#argmin
    mapper_1_output[data_cluster_key][0] += 1 # counter
    mapper_1_output[data_cluster_key][1][0] += data[0] #sum all SHOT_DIST
    mapper_1_output[data_cluster_key][1][1] += data[1] #sum all CLOSE_DEF_DIST
    mapper_1_output[data_cluster_key][1][2] += data[2] #sum all SHOT_CLOCK
  except:
   continue

combiner_1_input = mapper_1_output
for key, values in combiner_1_input.items():
 print('{key}\t{values}'.format(key=key, values=values))
```

Reducer 1: Finding the centroid of each cluster to update the new centroid in the next for loop which is implemented in the test.sh file.
The for loop inside the **test.sh** will stop when it finds that the centroids coming from Reducer 1 converge.

```
import sys
reducer_1_output = {}
for line in sys.stdin:
  key, values = line.split('\t')
  values = eval(values)
  count = values[0]
  sum_SHOT_DIST = values[1][0]
  sum_CLOSE_DEF_DIST = values[1][1]
  sum_SHOT_CLOCK = values[1][2]
  reducer_1_output[int(key)] = [sum_SHOT_DIST/count, sum_CLOSE_DEF_DIST/count,
sum_SHOT_CLOCK/count]
output = ''
for key, values in reducer_1_output.items():
 for value in values:
  output = output + '_' +str(value)
output = 'ClusterZ' + output[1:]
print(output)
```

Mapper 2: Outside the for loop to get the centroids, Mapper 2 filters the dataset just for the required players, calculating the euclidean distance of each data point from each of the 4 clusters and assigning it the key of the cluster with minimum distance. In this way we have 4 clusters for each player for the matrix {SHOT DIST, CLOSE DEF DIST, SHOT CLOCK} .

```
import sys
zones_mapper = sys.argv[1]
zones_mapper = [eval(dp) for dp in zones_mapper.split('Z')[1].strip('_').split('_')]
zones_mapper = {1:[zones_mapper[0],zones_mapper[1],zones_mapper[2]],
            2:[zones_mapper[3],zones_mapper[4],zones_mapper[5]],
            3:[zones_mapper[6],zones_mapper[7],zones_mapper[8]],
            4:[zones_mapper[9],zones_mapper[10],zones_mapper[11]]}
def euclidean_distance(A, B):
  return sum((a-b)**2 for a, b in zip(A[:], B[:])) ** (1/2)
players = ['stephen curry', 'james harden', 'chris paul','lebron james']
for line in sys.stdin:
  line = line.strip(',').split(',')
  line_len = len(line)
  player = line[-2]
  if line_len == 23 and player in players:
     try:
        player = player.split(' ')
        player = player[0]+player[1]
        shot = 1 if line[14] == 'made' else 0
        SHOT_DIST = float(line[12].strip(""))
        CLOSE_DEF_DIST = float(line[-5].strip(""))
        SHOT_CLOCK = float(line[9].strip(""))
        data = [SHOT_DIST, CLOSE_DEF_DIST, SHOT_CLOCK]
```

```
            data_centroids_distances = {1: euclidean_distance(data, zones_mapper[1]),
                                        2: euclidean_distance(data, zones_mapper[2]),
                                        3: euclidean_distance(data, zones_mapper[3]),
                                        4: euclidean_distance(data, zones_mapper[4])}
            data_cluster_key = min(data_centroids_distances, key = data_centroids_distances.get)
#argmin
            print(player +"_"+str(data_cluster_key)+"\t"+str(shot))
        except:
            continue
```

Reducer 2: Finding the count for the hit rate by counting the shots against each player within each cluster.

```
import sys
dict_score_count = {}
for line in sys.stdin:
   record = line.split('\t')
   data,count = record[0], record[1]
   try:
       count = int(count)
       dict_score_count[data] = [dict_score_count.get(data, [0,0])[0] + count,
dict_score_count.get(data, [0,0])[1]+1]
   except ValueError:
       pass
for key, value in dict_score_count.items():
   print(key+'\t'+str(value))
```

Mapper 3: Calculating hit rate for each player.

```
import sys
for entry in sys.stdin:
   player_cluster, result_shots = entry.split('\t')
```

```
    try:
        result_shots = eval(result_shots)
        if result_shots[1] == 1 and result_shots[0] == 0:
            continue
        else:
            hit_rate = result_shots[0]/result_shots[1]
    except:
        pass
    print(player_cluster+'\t'+str(hit_rate))
```

Reducer 3: Sorting and shuffling the output of the reducer to get the cluster with the highest hit rate.

```
import sys
reducer_3_output = {}
for entry in sys.stdin:
    player_cluster, hit_rate = entry.split('\t')
    player, cluster = player_cluster.split('_')
    hit_rate = float(hit_rate)
    if player not in reducer_3_output:
        reducer_3_output[player] = [cluster, hit_rate]
    elif hit_rate > reducer_3_output[player][1]:
        reducer_3_output[player] = [cluster, hit_rate]
    else:
        continue
for key, value in reducer_3_output.items():
    print(key+'\t'+'Cluster: '+str(value[0])+' | Hit Rate: '+str(value[1]))
```

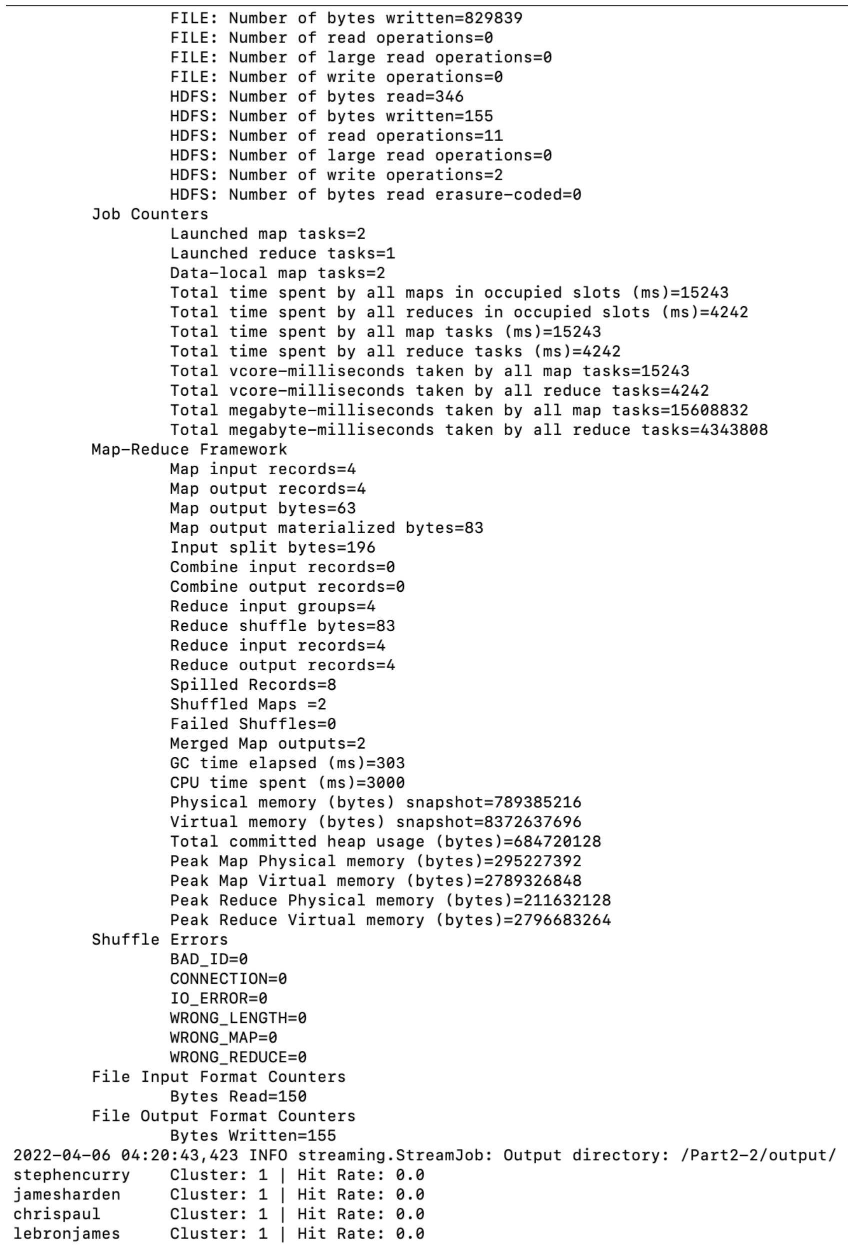
Test.sh: Only outputs the key value of all the four players', Stephen Curry, James Harden, Chris Paul, and Lebron James, the most comfortable zone/cluster with the highest hit rate, respectively. The following is what contains in our test.sh file:

```
#!/bin/sh

# starting_zones={1:[[$(( $RANDOM % 29 + 0 )),$(( $RANDOM % 11 + 0 )),$(( $RANDOM % 26 + 0
))]],2:[$(( $RANDOM % 29 + 0 )),$(( $RANDOM % 11 + 0 )),$(( $RANDOM % 26 + 0 ))],3:[$((
$RANDOM % 29 + 0 )),$(( $RANDOM % 11 + 0 )),$(( $RANDOM % 26 + 0 ))],4:[$(( $RANDOM %
29 + 0 )),$(( $RANDOM % 11 + 0 )),$(( $RANDOM % 26 + 0 ))]}
# Y=\'$(echo "$starting_zones" | sed "s/\[//; s/]//; s/'//g")\'
starting_zones=ClusterZ6_3_26_15_8_1_2_5_10_26_4_5
# $Y

zones="0"
new_zones=""

# ../../start.sh

for i in {0..50}; do
   if [[ $zones == $new_zones ]]; then
      break
   elif [[ $new_zones != "" ]]; then
      zones=$new_zones
   else
      zones=$starting_zones
   fi

   new_zones=`cat /BigData_Project/test-data\/shot_logs.csv | python3
/BigData_Project/part2/Part2/mapper1.py "$zones" | python3 /BigData_Project/part2/Part2/reducer1.py`
done

../../start.sh

/usr/local/hadoop/bin/hdfs dfs -rm -r /Part2/input/
/usr/local/hadoop/bin/hdfs dfs -rm -r /Part2/output/
/usr/local/hadoop/bin/hdfs dfs -mkdir -p /Part2/input/
/usr/local/hadoop/bin/hdfs dfs -copyFromLocal ../../test-data/shot_logs.csv /Part2/input/
/usr/local/hadoop/bin/hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar \
-file ../../part2/Part2/mapper2.py -mapper "../../part2/Part2/mapper2.py $zones" \
-file ../../part2/Part2/reducer2.py -reducer ../../part2/Part2/reducer2.py \
-input /Part2/input/* -output /Part2/output/

/usr/local/hadoop/bin/hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar \
-file ../../part2/Part2/mapper3.py -mapper ../../part2/Part2/mapper3.py \
-file ../../part2/Part2/reducer3.py -reducer ../../part2/Part2/reducer3.py \
-input /Part2/output/* -output /Part2-2/output/

/usr/local/hadoop/bin/hdfs dfs -cat /Part2-2/output/part-00000
/usr/local/hadoop/bin/hdfs dfs -rm -r /Part2/input/
/usr/local/hadoop/bin/hdfs dfs -rm -r /Part2/output/
/usr/local/hadoop/bin/hdfs dfs -rm -r /Part2-2/output/
../../stop.sh
```

Our main goal here is to create a for loop iterations of our clustering algorithm with a specific stop condition where we will compare old and new zones. Once we get a final zone it will be placed into two mapreduce via streaming.

The for loop inside the **test.sh** will stop when it finds that the centroids coming from Reducer 1 converge.

However, the following result is running it in the hadoop system:

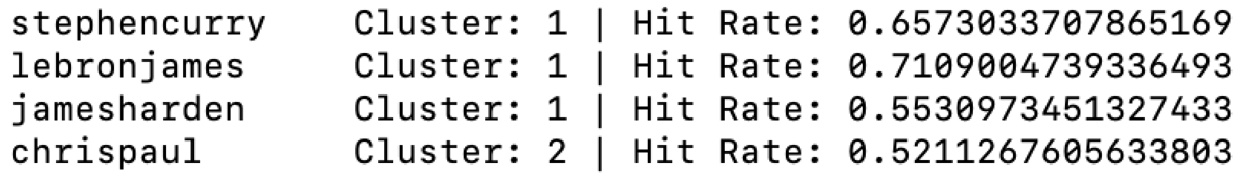


```
                FILE: Number of bytes written=829839
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=346
                HDFS: Number of bytes written=155
                HDFS: Number of read operations=11
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=15243
                Total time spent by all reduces in occupied slots (ms)=4242
                Total time spent by all map tasks (ms)=15243
                Total time spent by all reduce tasks (ms)=4242
                Total vcore-milliseconds taken by all map tasks=15243
                Total vcore-milliseconds taken by all reduce tasks=4242
                Total megabyte-milliseconds taken by all map tasks=15608832
                Total megabyte-milliseconds taken by all reduce tasks=4343808
        Map-Reduce Framework
                Map input records=4
                Map output records=4
                Map output bytes=63
                Map output materialized bytes=83
                Input split bytes=196
                Combine input records=0
                Combine output records=0
                Reduce input groups=4
                Reduce shuffle bytes=83
                Reduce input records=4
                Reduce output records=4
                Spilled Records=8
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=303
                CPU time spent (ms)=3000
                Physical memory (bytes) snapshot=789385216
                Virtual memory (bytes) snapshot=8372637696
                Total committed heap usage (bytes)=684720128
                Peak Map Physical memory (bytes)=295227392
                Peak Map Virtual memory (bytes)=2789326848
                Peak Reduce Physical memory (bytes)=211632128
                Peak Reduce Virtual memory (bytes)=2796683264
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=150
        File Output Format Counters
                Bytes Written=155
2022-04-06 04:20:43,423 INFO streaming.StreamJob: Output directory: /Part2-2/output/
stephencurry    Cluster: 1 | Hit Rate: 0.0
jamesharden     Cluster: 1 | Hit Rate: 0.0
chrispaul       Cluster: 1 | Hit Rate: 0.0
lebronjames     Cluster: 1 | Hit Rate: 0.0
```

Everything comes out as zeros, yet we will be using the following pipeline to get our desired output:

cat /BigData_Project/test-data\/shot_logs.csv | python3 /BigData_Project/part2/Part2/mapper2.py ClusterZ6.118314999217948_3.0200563057192005_20.761550492675013_14.78305407766219_4.3298 73739379016_6.644627967918709_5.413006608999867_2.5143065409660954_10.873875012148929_2 2.741569577169916_5.264388881616298_11.90512937993624 | python3 /BigData_Project/part2/Part2/reducer2.py | python3 /BigData_Project/part2/Part2/mapper3.py | python3 /BigData_Project/part2/Part2/reducer3.py

The following is the output coming from the test.sh:



```
stephencurry    Cluster: 1 | Hit Rate: 0.6573033707865169
lebronjames     Cluster: 1 | Hit Rate: 0.7109004739336493
jamesharden     Cluster: 1 | Hit Rate: 0.5530973451327433
chrispaul       Cluster: 2 | Hit Rate: 0.5211267605633803
```