THE UNIVERSITY OF SYDNEY

School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - GROUP ASSESSMENT

Unit of Study: COMP 5349 Cloud Computing

Assignment name: Hadoop MapReduce Programming

Tutorial time: Thursday 4p.m-6p.m **Tutor name:** Waiho Wong

DECLARATION
We the undersigned declare that we have read and understood the *University of Sydney Academic Dishonesty and Plagiarism in Coursework Policy*, an, and except where specifically acknowledged, the work contained in this assignment/project is our own work, and has not been copied from other sources or been previously submitted for award or assessment.

We understand that failure to comply with the *Academic Dishonesty and Plagiarism in Coursework Policy* can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

We realise that we may be asked to identify those portions of the work contributed by each of us and required to demonstrate our individual knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Project team members				
Student name	**Student ID**	**Participated**	**Agree to share**	**Signature**
1.Xuhong Guo	450489321	(Yes) / No	Yes/(No)	Xuhong
2.Yuchen Zhao	460134794	(Yes) / No	Yes / (No)	Yuchen zhao

SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.info@sydney.edu.au
sydney.edu.au/it

ABN 15 211 513 464 CRICOS 00026A

Assignment 1 Cloud Computing

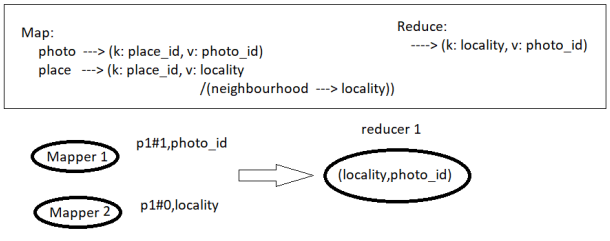
Xuhong Guo 450489321 *Yuchen Zhao* 460134794

Abstract. In this assignment We will write a series of Hadoop jobs to analyze a data set from flickr.com. The analysis is structured into three separate tasks which build on each other. We will practice basic Hadoop programming features and also observe the ease of use of the Hadoop framework. The data set is the same as those we have used in the labs.

1 Number of photos taken per locality.

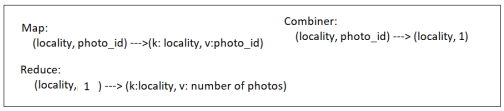
1.1 Mapreduce 1

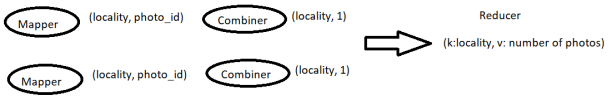
In order to find every locality photo numbers, we would use two map reduce in this question. In the first part map reduce, we would Implementing and partitioner in Hadoop, because Hadoop supports multiple inputs as well as different mappers, so we use different values from inputs. For the photo record, which comes from n0*.txt. We use place-id as the key followed by #1 to represent the source from and values are photo-id(key:place-id, value: photo-id). For the place record, we set place-id as the key with #0 and value is place-name from place.txt file(key:place-id,value:place-name).Use the reduce-side join by the key place-id, and final output is the values which is locality name and photo-is.We also set place-type-id 22 belongs to the place-type-id7 as the question requested.By the way, the characteristic are shorten of attributes' name in files.



1.2 Mapreduce 2

The second part map reduce in question one, we use the map to read result from previous map reduce. Then we use combiner function to get the photo number for each locality, reduce is used to sum all these photo numbers by each locality which can be shown in diagrams.





1.3 Total running time

The total running time in for question 1 is 5 minutes.

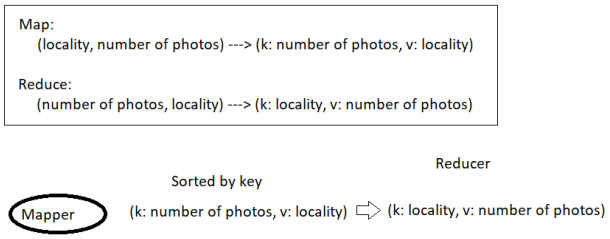
1.4 Result

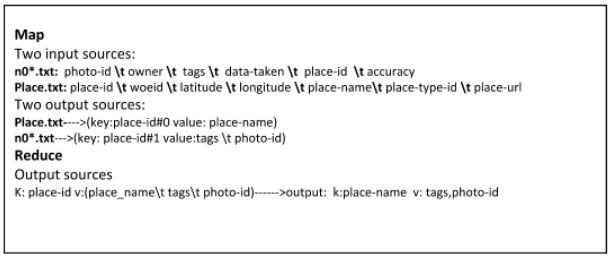
```
'Ain Taya, Boumerdes, Algeria   7
'Ain al 'Awda, Rabat-Sale, Morocco  1
'Ali Shah `avaz, Tehran, Iran   7
's-Graveland, North Holland, Netherlands    13
's-Gravenbrakel, Hainault, Belgium  33
's-Gravendeel, South Holland, Netherlands   4
's-Gravenpolder, Zeeland, Netherlands   11
's-Gravenweg, South Holland, Netherlands    19
's-Gravenwezel, Antwerp, Belgium    8
's-Gravenzande, South Holland, Netherlands  56
's-Heer Hendrikskinderen, Zeeland, Netherlands  16
's-Heer-Abtskerke, Zeeland, Netherlands 3
's-Heer-Arendskerke, Zeeland, Netherlands   3
```

2 The top 50 locality level places based on the number of photos taken in this locality.

2.1 Mapreduce

We could get the number of photos in each locality from the sector 1.Firstly we use map to get these source, which can be shown in table. For the reduce part, we set the a total list sized by 50,using reduce to list the top 50. Moreover, we use a reserve function here by the key. The reduce table could be seen in diagram as well.





2.2 Running Time

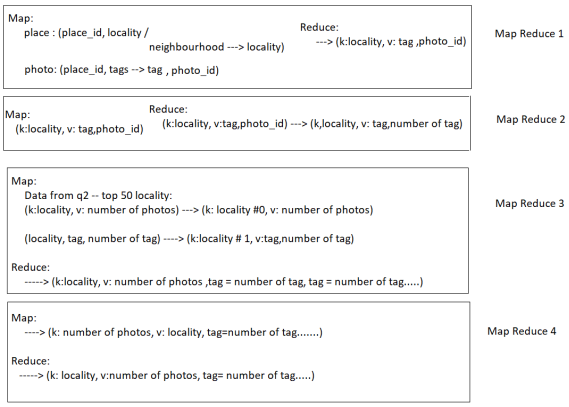
This running time is 1 minute.

2.3 Result

```
 4  New York, NY, US, United States 1188956
 5  Seattle, WA, US, United States  698078
 6  NY, US, United States   613379
 7  Chicago, IL, US, United States  566404
 8  Paris, Ile-de-France, FR, France    532534
 9  Washington, DC, US, United States   502270
10  Tokyo, Tokyo Prefecture, JP, Japan  498360
11  Taipei City, Taipei City, TW, Taiwan    494227
12  Sydney, NSW, AU, Australia  489161
13  Los Angeles, CA, US, United States  456481
14  Toronto, ON, CA, Canada 445834
15  Vancouver, BC, CA, Canada   429842
16  Melbourne, VIC, AU, Australia   352227
17  Tokyo, Tokyo Prefecture, Japan  342004
18  New York, NY, United States 318895
19  Austin, TX, US, United States   315899
20  Portland, OR, US, United States 281729
21  San Diego, CA, US, United States    272381
22  Barcelona, Catalonia, ES, Spain 265809
23  San Francisco, California, United States    242558
24  Boston, MA, US, United States   235339
```

3 Ten most popular tags for each of the top-50 localities.

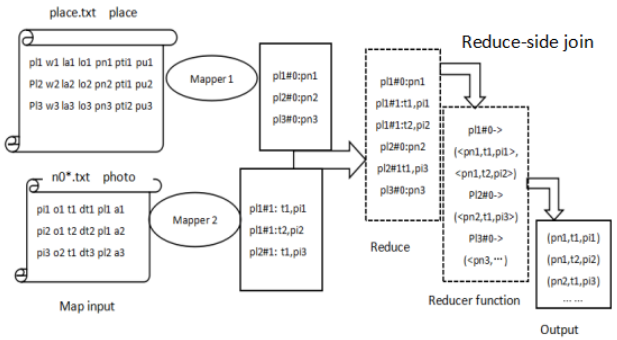
In this question, we plan to use four map reduce in Hadoop. The graph below shows the simple inputs and outputs of each map reduce process.



3.1 Mapreduce 1

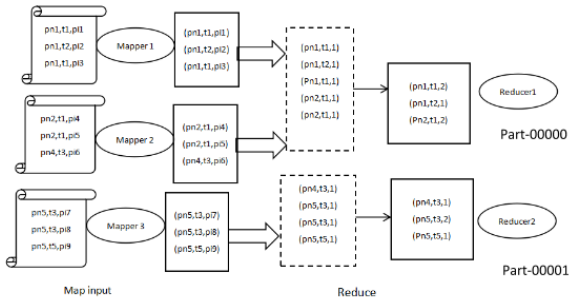
At first, we will explain the first map reduce structure. we could know the input and output of the first map reduce process.

In the first map reduce, we use the function partitioner to differ the data comes from two separate input sources based on the original key place-id. The key and value comes from place information is marked as #0, key and value about photo which comes from n0*.txt is marked as #1. Using reduce-side join by the same key which is place-id. we use reduce function to get the same key with different values.We get the key with place-id, value pairs with locality, tags and photo-id. Furthermore,we set the output of reduce is place-name, tags and photo-id. This changing is convenient for further using. The following diagram would show this process design well, however, the diagram shows simple example, due to large data-set, we set 7 map partitoners and 6 reduce partitoners.
The important point is that we filter out place or year tags in the reduce process here in void of useless tags, which including the lower characteristics' problems, same locality problems and/or data-time problems.



3.2 Mapreduce 2

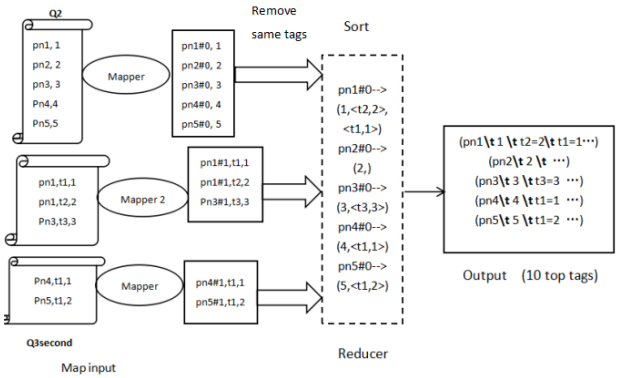
The second map reduce focuses on tags' processing. We use this map reduce to get each tags' number. In this process, map is used to read the information to lines. Reduce is used to count the same tags number under the same key. The output of this map reduce is locality, tags and number of this tag.



3.3 Mapreduce 3

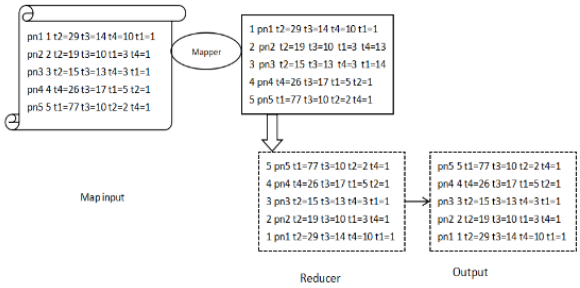
Map is used to get the previous output and input sources. In reduce process, we use keypartitioner to reduce-side join by placename. We set a dictionary to reduce the repeated tags. We set a total list sized by 10 to get the top ten tags and reversed by the tags' number decreasing.

Map
Two input sources:
Q2: photo-name **\t** number of photos
Q3second map reduce: place-name **\t** tags **\t** numbers
Two output sources:
Q2--->(key:place-name#0 value: number of photos)
Q3second map reduce--->(key: place-name#1 value:tags \t numbers)
Reduce
Output sources
K: place-name v:(number of photos, tag1\t number1, tag2\t number2…tag10\t number10)



3.4 Mapreduce 4

At last, we use a map reduce to process reserve, use -nr in reduce to make the results sorted by each places' photo numbers decreasing. Using map to read lines and using reduce to accomplish sorting. The following diagram shows the structure of fourth map reduce in this question.



3.5 Running Time

This running time is nearly 20 minutes.

3.6 Tags Cleaning

- Tag with year is deleted.
- Tag include london, uk when the place contains London, UK is deleted.
- Tag like sanfrancisco but the place is San Francisco is deleted.
- Tag list new or york, but the place is New York we keep it to prevent not the same meaning.

3.7 Result

4 Appendix

The HDFS location of our final output files from various executions.

4.1 Question 1

- /user/xguo8788/q1
- /user/yzha3522/q1

4.2 Question 2

- /user/xguo8788/q2
- /user/yzha3522/q2

4.3 Question 3

- /user/xguo8788/q3
- /user/yzha3522/q3