CSC 4760/6760 Big Data Programming
Spring 2020

All Students

Exam 1 - Solutions

02/26/2020

Name: ______________________________

Panther ID: ______________________________

Note: You may use additional white papers if the space is not enough.

This table is to be filled by the grader.

Problem	Total Score	Score
1	15	
2	30	
3	20	
4	5	
5	25	
6	5	
Total	100	

Problem 1. (Hadoop Commands) David is learning Hadoop. David has installed Hadoop in the Ubuntu virtual machine. Please help him with the following questions. Suppose that the full path of hadoop is:

/home/david/hadoop/

1.1 The command to start HDFS is

```
$ ~/hadoop/sbin/start-dfs.sh
```

1.2 David wants to take a look at the folders and files in HDFS. He should run the following command:

```
$ hdfs dfs -ls -R
```

1.3 Suppose there is already a folder "/user/david/" in HDFS. David wants to create a folder "/user/david/data/". The command is

```
$ hdfs dfs -mkdir /user/david/data
```

1.4 David has a file "peterpan.txt" in the local file system. The full directory is

/home/david/data/peterpan.txt

David wants to upload the data from local file system to the folder "/user/david/data/" in HDFS. The command is:

```
$ hdfs dfs -put /home/david/data/peterpan.txt /user/david/data/
```

1.5 The jar file is in the following folder in the local file system:

/home/david/data/WordCount.jar

David wants to run this .jar file on the file "/user/david/data/peterpan.txt" in HDFS. The output directory is "/user/david/data/out01". Please provide the command:

```
$ hadoop jar ~/data/WordCount.jar /user/david/data/peterpan.txt /user/david/data/out01
```

1.6 After hadoop runs successfully, David needs to see the output file. Please provide the command to see that:

```
$ hdfs  dfs  -cat  /user/david/data/out01/part-r-00000
```

1.7 David wants to transfer this output file from HDFS to the folder "/home/david/data" in the local file system. Please provide the command to do that.

```
$ hdfs  dfs -get  /user/david/data/part-r-00000    /home/david/data/
```

1.8 David can also run the .jar file on the file "/home/david/data/peterpan.txt" in local file system. The output folder is "/home/david/data/out02". The command is:

```
$ hadoop  jar  ~/data/WordCount.jar   file:///home/david/data/peterpan.txt
file:///home/david/data/out02
```

Problem 2. (WordCount Example) Please design the mapper and reducer for the WordCount problem.

WordCount problem: Given an input text file, count the frequency of each word in the file.

Please design the input and output <key, value> pairs for the Mapper and Reducer.

	Mapper	Reducer
Input	<null, one line text>	<word, list of frequency values>
Output	<word, 1>	<word, frequency value>

We get the "WordCount.jar" file by using the above design. It only contains Mapper and Reducer. There is no Combiner and Partitioner in this program.

Please illustrate how the WordCount.jar program works on the following dataset. This text file only contains one line.

<u>the sound sounds sound. it is the right right, right?</u>

Suppose we run the WordCount.jar program on this file. Suppose that the space symbol, ".", ",", and "?" are all used for separating the words. The outputs of the Mapper are

<the, 1>
<sound, 1>
<sounds, 1>
<sound, 1>
<it, 1>
<is, 1>
<the, 1>
<right, 1>
<right, 1>
<right, 1>

The outputs of the reducer are

<is, 1>
<it, 1>
<right, 3>
<sound, 2>
<sounds, 1>
<the, 2>

Are the results sorted alphabetically? When was it sorted?

Answer: Yes. The <K2, V2> are sorted between mapper and reducer.

Suppose the chuck size is 64MB. How many mappers and reducers are used? Why?

Answer: 1 and 1. Since the input file size is less than 64 MB, we only have one mapper. Without specifying the number of reducers, the default number of reducer is 1.

Suppose we want to use 2 reducers. How to do that in the ".java" file?

job.setNumReduceTasks(2);

Suppose the key-value pairs with the keys "the", "sound", "is" are processed by reducer 1. Suppose the key-value pairs with the keys "it", "right", "sounds" are processed by reducer 2. Please determine the outputs of the two reducers.

Reducer 1	Reducer 2
<is, 1>	<it, 1>
<sound, 2>	<right, 3>
<the, 2>	<sounds, 1>

From the above results, we can see that

1. Are the records in the output file of one reducer sorted? Answer: Yes

2. Are the records in the output file of Reducer 1 greater alphabetically than those in the output file of Reducer 2? Answer: No.

If we want that the answer to the second question is "Yes", how can we design MapReduce program?

Answer: We can use partitioner.

Please design your partitioner with boundary:

If the first letter of a word is less than "s", this word belongs to partition 1; otherwise, this word belongs to partition 2. What are the partitions in the above example?

Partition 1	<it, 1>, <is, 1>, <right, 1>, <right, 1>, <right, 1>
Partition 2	<the, 1>, <sound, 1>, <sounds, 1>, <sound, 1>, <the, 1>

What are the outputs of reducer 1 and 2?

Reducer 1	Reducer 2
<is, 1>	<sound, 2>
<it, 1>	<sounds, 1>
<right, 3>	<the, 2>

Suppose we add the combiner to the program. Can we use the code for the reducer as the combiner?

Answer: Yes.

Will the combiner helps improve the computing efficiency and why?

Answer: Yes, since the combiner will combine the intermediate <K2, V2> pairs and save the communication costs. The data to be transmitted over the network will be reduced during the shuffling stage and the total computational time will be reduced.

How many combiners do we have?

Answer: 1. Please Indicate the output of the combiner.

<is, 1>
<it, 1>
<right, 3>
<sound, 2>
<sounds, 1>
<the, 2>

Problem 3. (Number of Mappers and Reducers) Suppose we have a cluster with 240 rack servers, and each rack server has 4 cores. We want to run “WordCount.jar” on a text file. Suppose the size of the input file is 36,853 MB. The chunk size in the HDFS is 64MB. Please answer the following questions.

1. How many mappers will be created when you run a Hadoop program on this cluster?

Answer: Approximately 36,853 MB / 64MB = 576 mappers. Hadoop will determine the number of mappers automatically.

2. Can we set the number of reducers to 0.9 * 240 * 4 = 864? If so, how many reducers will run simultaneously?

Answer: Yes, all the 864 reducers should run simultaneously.

3. Can we set the number of reducers to 1.7 * 240 * 4 = 1632? If so, how many reducers will run simultaneously?

Answer: Yes, approximately 240 * 4 = 960 reducers should be able to run simultaneously. The remaining (1632 – 960 = 672) reducers have to wait until some reducers finish their jobs and there are idles cores.

4. Describe the differences of the two strategies for setting the number of reducers? One is 864, the other is 1,632. Which one is better and why?

Answer: With 0.9, all of the reduces can launch immediately and start transferring map outputs as the maps finish. With 1.7 the faster nodes will finish their first round of reduces and launch a second wave of reduces doing a much better job of load balancing. Increasing the number of reduces increases the framework overhead, but increases load balancing and lowers the cost of failures.

5. When we set the number of reducers to 864, how many partitions are there in the program? How about 1,632 reducers?

Answer:

864. Since the number of partitions should be the same as the number of reducers.

1,632. Since the number of partitions should be the same as the number of reducers.

6. Suppose we use “RandomPartitioner” and we set the number of reducers to 864. Then how many partitions do we have?

Answer: 864. The number of partitions must be the same as the number of reducers.

Problem 4. (Passing Parameters)

In the top-k query problem, we need to pass the parameter k from the command line to the mapper or reducer. Please describe how to pass the parameters in Hadoop program.

Answer:

In the main() function:

```
Configuration conf = new Configuration();

conf.set("k value for top-k query", otherArgs[0]);
```

In the map() or reduce() functions:

```
Configuration conf = context.getConfiguration();

Long nKValue = Long.parseLong(conf.get("k value for top-k query"));
```

We use Counters in Hadoop to pass the parameter.

Problem 5. (PageRank)

The math equation of one page j for PageRank is:

$$r(\mathrm{Page}_j) = c \sum_{\mathrm{Page}_i \in I_{\mathrm{Page}_j}} \frac{r(\mathrm{Page}_i)}{|O_{\mathrm{Page}_i}|} + \frac{(1-c)}{n}$$

where $r(\mathrm{Page}_j)$ represents the PageRank value of Page_j, I_{Page_j} represents the set of pages pointing into Page_j, $|O_{\mathrm{Page}_i}|$ represents the number of outlinks (out-neighbors) from Page_i, $c \in [0,1]$ represents the decay factor, and n represents the number of web pages in the web graph. The decay factor is usually set to 0.85, i.e., $c = 0.85$.

If we write out the equations for all the nodes, we have a system of linear equations. Power iteration method is usually used to solve the system of linear equations. The time complexity is $O(\beta m)$, where β represents the number of iterations, and m represents the number of edges in the graph.

To design the MapReduce algorithm for implementing the power iteration method for computing PageRank, we need to first figure out the Mapper and Reducer.

Let us use O_i to represent the set of outlinks (out-neighbors) from Page_i and I_j to represent the set of inlinks (in-neighbors) pointing into Page_j.

Each mapper will read one line of the adjacency list file. That means each mapper knows all the outlinks (out-neighbors) of a node.

Mapper:

Input	PageRank value r_i^{t-1} of node i Outlinks O_i of node i (one row of the adjacency matrix)
Output	For each node $j \in O_i$, output < node j, $r_i^{t-1} \cdot \frac{1}{|O_i|}$ >

Each mapper processes a single node i. Each mapper knows the outlinks of node i. For each out neighbor node $j \in O_i$, the mapper will output a key-value pair, where the key is the index of node j, and the value is the value $r_i^{t-1} \cdot \frac{1}{|O_i|}$.

Reducer:

Input	<node j , a list of values $r_i^{t-1} \cdot \frac{1}{\lvert O_i \rvert}$ > , where node $i \in I_j$
Output	< node j, new PageRank value r_j^t >, where $r_j^t = c \sum_{i \in I_j} r_i^{t-1} \cdot \frac{1}{\lvert O_i \rvert} + \frac{(1-c)}{n}$

Each reducer processes a single node j. During the shuffle processes, all the key-value pairs with the same key will be sent to the same reducer. Therefore, each reducer knows the inlinks (in-neighbors) of node j. For each in-neighbor node $i \in I_j$, the reducer knows the value $r_i^{t-1} \cdot \frac{1}{\lvert O_i \rvert}$. The reducer will sum these values together. The reducer will then calculate the value of the PageRank equation $r_j^t = c \sum_{i \in I_j} r_i^{t-1} \cdot \frac{1}{\lvert O_i \rvert} + \frac{(1-c)}{n}$. The reducer will output a key-value pair, where the key is the index of node j, and the value is the value $r_j^t = c \sum_{i \in I_j} r_i^{t-1} \cdot \frac{1}{\lvert O_i \rvert} + \frac{(1-c)}{n}$.

Since the mapper needs to read two files, therefore we use the distributed cache technique to read the PageRank value file, and use the InputFileFormat to read one line in the adjacency list file.

The MapReduce program will repeat this process many times in order to calculate the final PageRank values. After each iteration, the intermediate PageRank values will written into the disk (HDFS, file system). At the beginning of each iteration, the program will read the intermediate PageRank values from the disk (HDFS, file system).

Can we use combiner in the program? Yes. If so, can we use the code of reducer as the combiner? No. If not, please describe the idea of combiner.

Combiner:

Input	<node j , a list of values x_i^{t-1} > , where node $i \in I_j$
Output	< node j, value y_i^t >, where $y_i^t = \sum_{i \in I_j} x_i^{t-1}$

Do we need to change the code of reducer? No.

Problem 6. (Skewed Distribution)

In the WordCount example, we have the skewed distribution problem. The frequency of some commonly used words are much higher than that of barely used words. The tasks for processing some keys take significantly longer time than other tasks.

Please briefly describe your idea for solving the skewed distribution problem such that the work loads are balanced across all the reducers.

Answer:

Combiner definitely will help. Combiner works locally on the output intermediate key-value pairs from Mapper and summarize them and output concise intermediate key-value pairs.

Partitioner can also be used to solve the skewed distribution problem. If you want to use Partitioner to solve the skewed distribution problem, you need to somehow know the prior knowledge. We can use random sampling to sample a small dataset and estimate the word frequency. Suppose we know that “hello” is frequent. Then in the custom partitioner function, we emit multiple hashcodes for “hello”. In the default partitioner, it will only emit one hashcode for “hello”. That means only one reducer will process “hello”. But if you customize it, we will have multiple reducers to process “hello”.