

CSC 555 Assignment 3

Sarah Cummings

1) From Mining Massive Data Sets:

a) Design Map Reduce algorithms to take a very large file and produce as output:

i. The same set of integers but with each integer only appearing once:

The map would take the integers as input and produce the integer as the key, and a one or blank as the value. Then the reducer would filter by key and produce a list of distinct keys, which would be our set of integers with each only appearing once.

ii. The count of the number of distinct integers in the input:

We could continue with the file produced in the last question, and have a new mapper return each distinct integer as key, and a 1 as the value. Then a reducer would sum up all the values, which would provide a count of our distinct values.

b) Exercise 2.4.1 : Suppose a job consists of n tasks, each of which takes time t seconds. Thus, if there are no failures, the sum over all compute nodes of the time taken to execute tasks at that node is nt . Suppose also that the probability of a task failing is p per job per second, and when a task fails, the overhead of management of the restart is such that it adds $10t$ seconds to the total execution time of the job. What is the total expected execution time of the job?

Can this be thought of as a Bernoulli Distribution Problem? Or is this a geometric distribution problem?

Given p = probability of failure, n as the number of tasks, and k as the number of possible outcomes, we have the following equation to represent the number of failures:

$$p(k) = \binom{n}{k} (1-p)^{n-k} p^k$$

However we could have this number of failures per task per second, so we must multiply by t . And then for every failure, we have the add time of $10t$ so we multiply by $10t$.

Then finally, we have the perfect success attempt that takes time nt , so we must add that to our equation.

So the total expected time is

$$\left(\binom{n}{k} (1-p)^{n-k} t * 10t \right) + nt$$

c) What is the communication cost of each of the following algorithms, as a function of the size of the relations matrices or vectors of which they are applied? Note: communication cost of a task is the size of the input to the task. The communication cost of an algorithm is thus the sum of all communication costs of all the tasks implementing the algorithm.

ii. The union algorithm of 2.3.6 (Map tasks take chunks of one of two relations, R and S, with same schema and produce each tuple into key-value pair. Reducer will produce output of all of the tuples):

With this union function, the mapper processes key-value pairs for every entry in R and S so the communication cost is the total number of entries in R and the total number of entries in S.

iii. The aggregation algorithm of 2.3.8 (mapper takes tuples (a, b, c) from relation

R and produces key-value pair (a, b):

The communication cost for this task is just the number of tuples, (a, b, c) in R.

2) Suppose you are tasked with analysis of the company's web server logs. The log dump contains a large amount of information (a number of different properties recorded for every access) – and you regularly run a Hadoop job that analysis the entire file. Come up with two different possible suggestions that could improve performance of the query.

One of the ways we could improve performance of the query is by adding more worker nodes. If the file is very large, with many blocks of data, having more mappers or reducers could reduce the overall runtime and improve performance.

We could also add a combiner if we do not have one already. This would summarize the map output records with the same key and reduce the volume of data transferred from the mapper to the reducer.

3)

a) Suppose you have a 6-node cluster with replication factor of 3. Describe what HDFS has to do after it determines that a node has crashed.

If the cluster has a replication factor of three, that means that the 6-node cluster is replicated in 3 different locations. When one of the nodes has been determined to have crashed, the master node will note which node has failed and run the required remaining calculation task on another replication of the node.

b) If a mapper task fails due to a crash of the current node, which other node(s) is the best place to re-run the same task?

Hadoop creates mapper task for each split chunk. Hadoop generally tries to run the map task on the node where the input data or input split resides in, which improves performance via data locality optimization. If the current node crashes, the system will run on one other other nodes where the input replica is located.

4) Input data file of size equal to 25 disk blocks, with replication factor set to one. The mapper in this job requires 2 minutes to read and fully process a single block of data, and reducer takes no time.

a) Approximately how long will it take to process the file if you only had one Hadoop worker node? You can assume that that only one mapper is created on every node.

50 minutes— two minutes to process each block.

b) 15 Hadoop worker nodes?

4 minutes— each of the 15 worker nodes process one block of the 25 which takes 2 minutes, but then there are 10 more blocks to process. 10 of the worker nodes will process those remaining 10 blocks in 2 more minutes.

c) 100 Hadoop worker nodes?

2 minutes— 25 of the worker nodes each process one of the blocks, which takes 2 minutes.

d) Now suppose you were told that the replication factor has been changed to 3?

That is, each block is stored in triplicate, but file size is still 25 blocks. Which of the answers (if any) in a)-c) above will have to change?

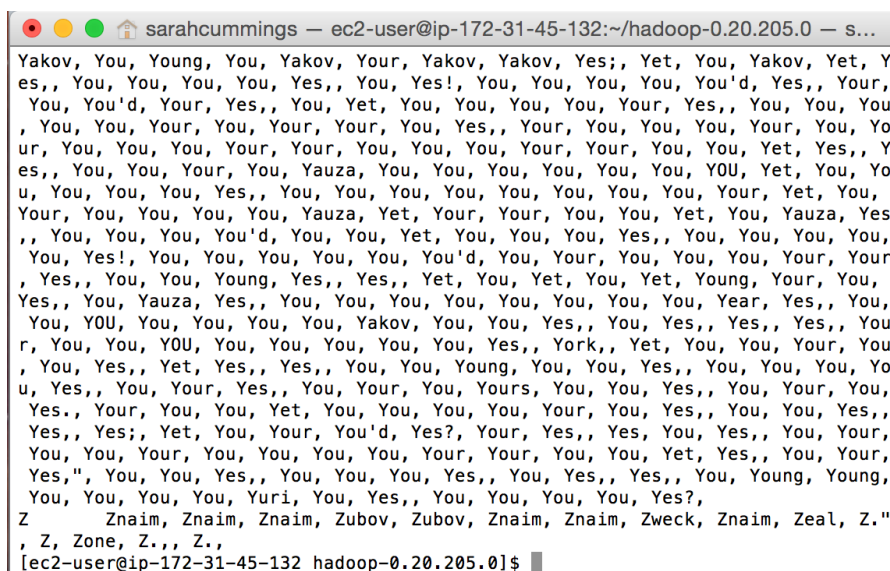
None of the answers will change. The only reason our time would increase is if there is failure in our node, at which point we would make use of another replication of the data.

- 5) b) The line for key Z contains: Z Znaim, Znaim, Znaim, Zubov, Zubov, Znaim, Znaim, Zweck, Znaim, Zeal, Z.", Z, Zone, Z., Z.,

The run time was as follows:

```
real    0m39.959s
user    0m1.592s
sys     0m0.064s
```

This is the output:



```
sarahcummings - ec2-user@ip-172-31-45-132:~/hadoop-0.20.205.0 - s...
Yakov, You, Young, You, Yakov, Your, Yakov, Yakov, Yes;, Yet, You, Yakov, Yet, Y
es,, You, You, You, You, Yes,, You, Yes!, You, You, You, You, You'd, Yes,, Your,
You, You'd, Your, Yes,, You, Yet, You, You, You, You, Your, Yes,, You, You, You
, You, You, Your, You, Your, Your, You, Yes,, Your, You, You, You, Your, You, Yo
ur, You, You, You, Your, Your, You, You, You, Your, Your, You, You, Yet, Yes,, Y
es,, You, You, Your, You, Yauza, You, You, You, You, You, You, YOU, Yet, You, Yo
u, You, You, Yes,, You, You, You, You, You, You, You, Your, Yet, You,
Your, You, You, You, You, Yauza, Yet, Your, Your, You, You, Yet, You, Yauza, Yes
,, You, You, You, You'd, You, You, Yet, You, You, You, Yes,, You, You, You, You,
You, Yes!, You, You, You, You, You, You'd, You, Your, You, You, You, Your, Your
, Yes,, You, You, Young, Yes,, Yes,, Yet, You, Yet, You, Yet, Young, Your, You,
Yes,, You, Yauza, Yes,, You, You, You, You, You, You, You, Year, Yes,, You,
You, YOU, You, You, You, You, Yakov, You, You, Yes,, You, Yes,, Yes,, Yes,, You
r, You, You, YOU, You, You, You, You, You, Yes,, York,, Yet, You, You, Your, You
, You, Yes,, Yet, Yes,, Yes,, You, You, Young, You, You, Yes,, You, You, You, Yo
u, Yes,, You, Your, Yes,, You, Your, You, Yours, You, You, Yes,, You, Your, You,
Yes,, Your, You, You, Yet, You, You, You, Your, You, Yes,, You, You, Yes,,
Yes,, Yes;, Yet, You, Your, You'd, Yes?, Your, Yes,, Yes, You, Yes,, You, Your,
You, You, Your, You, You, You, Your, Your, You, You, Yet, Yes,, You, Your,
Yes,", You, You, Yes,, You, You, You, Yes,, You, Yes,, Yes,, You, Young, Young,
You, You, You, You, Yuri, You, Yes,, You, You, You, You, Yes?,
Z
Z
Znaim, Znaim, Znaim, Zubov, Zubov, Znaim, Znaim, Zweck, Znaim, Zeal, Z."
, Z, Zone, Z.,, Z.,
[ec2-user@ip-172-31-45-132 hadoop-0.20.205.0]$
```

d) Edit the file to add the combiner:

```
sarahcummings — ec2-user@ip-172-31-45-132:~/hadoop-0.20.205.0 — s...
```

```
GNU nano 2.3.1      File: myNewJar/CollectWords.java

conf.setOutputValueClass(Text.class);

conf.setMapperClass(Map.class);
conf.setCombinerClass(Reduce.class);
conf.setReducerClass(Reduce.class);

conf.setInputFormat(TextInputFormat.class);
conf.setOutputFormat(TextOutputFormat.class);

FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));

JobClient.runJob(conf);
}
```

```
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

The line for Z contains: Z Znaim, Znaim, Znaim, Zubov, Zubov, Znaim, Znaim, Zweck, Znaim, Zeal, Z.", Z, Zone, Z., Z.,

The runtime was as follows:

```
real    0m1.441s
user    0m1.336s
sys     0m0.068s
```

This was the output:

```

[ec2-user@ip-172-31-45-132:~/hadoop-0.20.205.0 - s...
Yakov, You, Young, You, Yakov, Your, Yakov, Yakov, Yes;, Yet, You, Yakov, Yet, Y
es,, You, You, You, You, Yes,, You, Yes!, You, You, You, You, You'd, Yes,, Your,
    You, You'd, Your, Yes,, You, Yet, You, You, You, You, Your, Yes,, You, You, You
    , You, You, Your, You, Your, Your, You, Yes,, Your, You, You, You, Your, You, Yo
ur, You, You, Your, Your, You, You, You, Your, Your, You, You, Yet, Yes,, Y
es,, You, You, Your, You, Yauza, You, You, You, You, You, You, YOU, Yet, You, Yo
u, You, You, You, Yes,, You, You, You, You, You, You, You, Your, Yet, You,
    Your, You, You, You, Yauza, Yet, Your, Your, You, You, Yet, You, Yauza, Yes
    ,, You, You, You, You'd, You, You, Yet, You, You, You, Yes,, You, You, You, You,
    You, Yes!, You, You, You, You, You, You'd, You, Your, You, You, Your, Your
    , Yes,, You, You, Young, Yes,, Yes,, Yet, You, Yet, You, Yet, Young, Your, You,
    Yes,, You, Yauza, Yes,, You, You, You, You, You, You, You, Year, Yes,, You,
    You, YOU, You, You, You, You, Yakov, You, You, Yes,, You, Yes,, Yes,, Yes, You
r, You, You, YOU, You, You, You, You, You, Yes,, York,, Yet, You, You, Your, You
    , You, Yes,, Yet, Yes,, Yes,, You, You, Young, You, You, Yes,, You, You, You, Yo
u, Yes,, You, Your, Yes,, You, Your, You, Yours, You, You, Yes,, You, Your, You,
    Yes,, Your, You, You, Yet, You, You, You, Your, You, Yes,, You, You, Yes,,
    Yes,, Yes;, Yet, You, Your, You'd, Yes?, Your, Yes,, Yes, You, Yes,, You, Your,
    You, You, Your, You, You, You, You, Your, Your, You, You, Yet, Yes,, You, Your,
    Yes,, You, You, Yes,, You, You, Yes,, You, Yes,, Yes,, You, Young, Young,
    You, You, You, You, Yuri, You, Yes,, You, You, You, You, Yes?,
Z      Znaim, Znaim, Znaim, Zubov, Zubov, Znaim, Znaim, Zweck, Znaim, Zeal, Z."
    , Z, Zone, Z,, Z,,
[ec2-user@ip-172-31-45-132 hadoop-0.20.205.0]$ █

```

Note the output and final line are the same but the run time is shorter with the combiner.

e) Did you expect the output and the time-to-run in b) to be the same as the time and output in c) above?

I expected the combiner to shorten the runtime for this MapReduce job.

f) Why?

The main purpose of a combiner is to summarize the map output records with the same key. This reduces the volume of data transferred from the mapper to the reducer and thus cuts the amount of time needed to complete the task.

- 6) a) Compute the cost of provisioning 1 Petabyte using SSD storage for a duration of one month (e.g., <https://aws.amazon.com/ebs/pricing/>). You can assume General Purpose SSD.

Amazon EBS general purpose SSD (gp2) cost \$0.10 per GB/month of storage.
1 petabyte = 1,000,000 gigabytes.

So the cost is $\$0.10 * 1,000,000 = \$100,000$

b) Compute the cost of provisioning one Petabyte of storage using magnetic drives (there is a link on the bottom to the "Previous Generation Volume" page)

For Amazon EBS magnetic volumes, the cost is \$0.05 per gigabyte, \$0.05 per 1 million I/O requests.

Since the price is half of what it costs in part A, the total cost is half: \$50,000