Assignment 1: Basics and Map-Reduce

1. Exercise 1

   Question 1:

   Answer: $2.25 * 10^{6}$

   Explanation: only the days is changed so equation should be:

   $5 * 10^{17} * 4.5 * 10^{5} * 10^{-18} = 2.25 * 10^{6}$

   Question 2:

   Answer: $2.5 * 10^{6}$

   Explanation: number of people and hotels are changed so equation should be:

   $9/2 * 10^{18} * 5 * 10^{5} * 1/9 * 10^{-18} = 2.5 * 10^{6}$

   Question 3:

   Answer: $2*10^{-8}$

   Explanation: the days changed so the equation should be:

   $5 * 107 * \binom{1000}{4} * 10\text{-}36 \approx 2 * 10^{-8}$

2. Exercise 2

Summary of 2.4:

2.4 extensions to MapReduce

There are many systems have been spawned from MapReduce. These methods have many common characteristic, and they create a method to deal with the failures in execution.

Two method called clustera and hyracks extend the two step workflow to collection of functions. This method uses an arcs a->b to represent the a' s output is the input of b. Master controller will divide the tasks during the execution process. The function only delivers the output after they complete, so there is no need for worrying duplicate output. The advantage of this method is that all the task will be divided by master controller to eliminate the distributed file system.

Many large-scale computations are recursions, and the computations means that computation of fixed-point of matrix-vector multiplication. Recursions are often implement by iterated MapReduce. There are two tasks, join tasks and duo-elim tasks. For n files, there are n join tasks, and each of these task corresponds to a function. There also m dup-elim task corresponds to function taking two arguments. One of the feature is that the tasks store their previously input, so we can deal with single node and single rack failure.

The last approach to deal with failures is pregel system. In this method, data is viewed by graph, and tasks are represented by nodes, and each node generate output and pass it to other nodes. To against the failure in computation, pregel checkpoints after supersteps. The checkpoint is consisting of copy of each task, so when failure occurs, the job can restart from recent point. This method sometimes may cause many normal tasks to redo their work, but it is acceptable.

Summary of 2.5:

2.5 the communication cost model

the communication cost model is a model to measure the quality of algorithms implemented in computing cluster. Moving data between tasks is bottleneck for most of the application.

The communication cost size is the input of task, so the communication cost of an algorithm is the sum of cost of all tasks. There are three aspects to explain the importance of communication cost. First, the execution time of algorithm is increasing by the size of input. Secondly, the communication speed is very slow when comparing to processor speed. Third, move data into main memory take more time. There are two reasons why do not take output of task into account. First, the output of a maybe the input of b, so there is no need to count the size of output. Secondly, after processing by algorithm, the output usually to be very small when comparing to the input.

Wall-clock time is the time that it takes a parallel algorithm to finish. For some reason, to minimize the communication cost, all the work may be assigned to one task, but in this situation, the wall-clock time will be very high.

There are four step to examine how the communication cost can be helpful in choosing an algorithm.

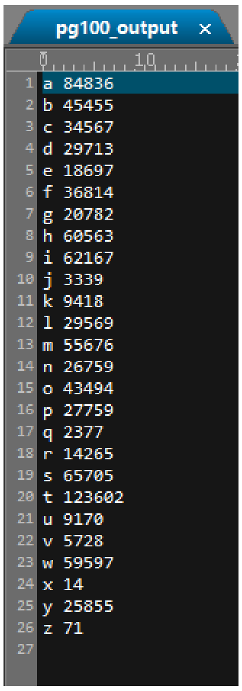
- Select attributes involved in the natural join
- Select buckets for each of these attributes
- Identify each of k reducer with a vector of bucket number
- Send tuples of relation to all those reducer

Take R(A,B)∞S(B,C)∞T(C,D) as an example, R-tuple and S-tuple agree on B, and S-tuple and T-tuple agree on C.

- Join the R and S first, and join with T, the communication cost is (r+s+t+prs).
- Join the S and T first, and join with R, the communication cost is (r+s+t+pst)
- The third way is joining these relation at once.

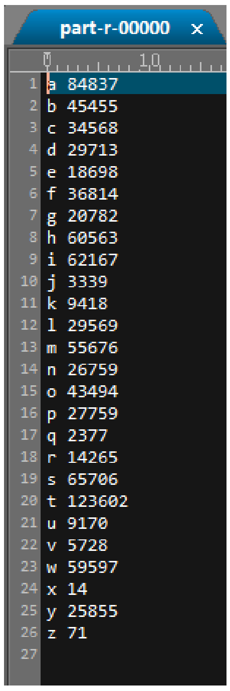
3. Exercise 3

Result of pg100.txt:



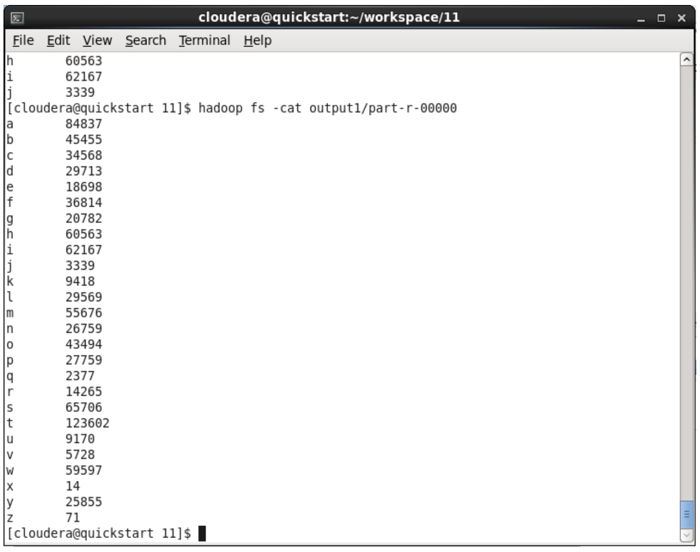
```
a 84836
b 45455
c 34567
d 29713
e 18697
f 36814
g 20782
h 60563
i 62167
j 3339
k 9418
l 29569
m 55676
n 26759
o 43494
p 27759
q 2377
r 14265
s 65705
t 123602
u 9170
v 5728
w 59597
x 14
y 25855
z 71
```

result of 100.txt in standalone mode



```
a 84837
b 45455
c 34568
d 29713
e 18698
f 36814
g 20782
h 60563
i 62167
j 3339
k 9418
l 29569
m 55676
n 26759
o 43494
p 27759
q 2377
r 14265
s 65706
t 123602
u 9170
v 5728
w 59597
x 14
y 25855
z 71
```

result of 100.txt in pseudo-distributed mode


cloudera@quickstart:~/workspace/11
File Edit View Search Terminal Help

```
h       60563
i       62167
j       3339
[cloudera@quickstart 11]$ hadoop fs -cat output1/part-r-00000
a       84837
b       45455
c       34568
d       29713
e       18698
f       36814
g       20782
h       60563
i       62167
j       3339
k       9418
l       29569
m       55676
n       26759
o       43494
p       27759
q       2377
r       14265
s       65706
t       123602
u       9170
v       5728
w       59597
x       14
y       25855
z       71
[cloudera@quickstart 11]$
```

4. Exercise 4

Some recommendation:

924	15416,6995,43748,45881,2409,11860,439
8941	8943,8944,8940
8942	8939,8940,8943,8944
9019	9022,9023,317
9020	9021,9022,9017,9016,9023,317
9021	9020,9022,9017,9016,9023,317
9022	9021,9020,9019,9023,317,9017,9016
9990	34642,13478,34299,37941,13877,34485,13134
9992	9989,9987,35667,9991
9993	9991,34642,13478,34299,37941,13877,34485,13134

Core idea: if user A have friend B and C, then B and C may be the recommendation to each other. At the same time A and B shouldn’ t be recommending and so does A and C.

Algorithm:

In the map part, the data is put into this kind of structure:

[user, recommend friend, mutual friend]

or [user, friend, -1]

In the reduce part will add up a number for each recommend friend if they are recommended for several times. If the recommend friend is already a friend of the user, they set the number to -1. After doing all of this, sort it and output the first 10 recommend friends. We use hashmap to store the recommend friends and their recommend time numbers.