CSE 512 Project Phase 3

Ziming Dong
Arizona State University
zdong27@asu.edu

Jianlun Li
Arizona State University
jianlunl@asu.edu

Chang Xu
Arizona State University
changxu6@asu.edu

1 ABSTRACT

This project phase is to run the experiments based on the code we developed in project one and project two. We test functions and operations and record the corresponding runtime cost, memory usage and communication cost. There are two parameters on these test that we can control: number of machines and size of datasets. The purpose of recording results from different tests is to generate the evaluation metrics and analysis the result.

2 KEYWORDS

Runtime, Memory usage, Communication cost, Machines, Dataset

3 INTRODUCTION

Since Hadoop and Apache Spark can help us set up more than one nodes to distribute task to each nodes so job will require less time to finish. Based on the previous experiences of implement Hadoop program, we use range query, rang join query, distance query, distance join query, hotzone analysis and hot cell analysis to complete jobs, however, we did not focus on the running cost instead of correct result. Now, we will test on two parameters for each job and record the evaluation metric for each test, we should be able to explore more findings based on the different experiments results.

4 EXPERIMENTAL SETUP

To run experiments, we set up three machines on the Amazon AWS, one master node and two slave nodes.Master nodes connects with two slaves, but slaves do not connect with each other.

The machine configuration for Master node:
Amazon EC2, t2.medium
Architecture: i386, x86-64
Memory: 4096MB
CPUs: 2
Cores: 2

The machine configuration for slave node:
Amazon EC2, t2.micro
Architecture: i386, x86-64
Memory: 1024MB
CPUs: 1
Cores: 1

5 EXPERIMENTAL EVALUATION

The dataset we are using to test are:
arealm10000.csv
zcta10000.csv
zone-hotzone.csv
yellow-trip-sample100000.csv
point-hotzone.csv

We develop python code to measure the runtime/memory/usage/number of bytes. We use the python inside function psutil to measure the, we create a new thread and to run the script.

Our group memeber Chang Xu developed two python files to assist to measure the performance, one is called pj3.py,pj3.py is the main test script of this phase. There are two experiments, one is measuring metrics by changing the size of the dataset, and another is by changing the number of machines. In pj3.py, task0() is for the fronter experiment, and task1() is for the latter. Both of the two functions call measure(), which runs commands for each of the six queries: ["rangequery", "rangejoinquery", "distancequery", "distancejoinquery", "hotzoneanalysis", "hotcellanalysis"]. pj3.py uses system argv as input, argv[1] represents the choice of which task to do. pj3.py redirects the output to a file named "pj3-log.log", which includes all the metrics such as runtime.pj3.py also uses os.system() to run commands. All the commands are run in new threads, with the main thread measure the metrics, especially memory usage, of the corresponding query. The memory usage is obtained by observing the maximum memory usage of the query process. Runtime and communication cost are measured by using the values after the process subtract the values before the process. All the file paths, query names and other constant parameters are HARD CODED in the code. Please change the corresponding parts in the code if you want to run the code on your machine.

The other is called csvpartitioner.py:
csvpartitioner.py serves as an tool function to partition an input file into user specified sizes.It takes system argvs as inputs, argv[1] is the input file path, and argv[2] is the partitioning size. The output file name is determined by these two argvs.

Runtime: we set up beginning time and ending time to calculate the runtime of processing.

Memory Usage: When script runs to the new thread, main thread keep loopping and get the current memory usage. When script run reach the end, we can find the maximum value of memory usage.

Communication cost: we measure the network IO before and after the process, then calculate the differences as the communication cost.

5.1 Different Number of Machine

We are testing based on same size of data which is medium but different number of machines
Three test results about varying number of machines shows below:
One Machine(master node):
Note: Because of this is one machine test, so we don't have communication cost for this test, however, due to the local effect from the local network, we can not get the exact communication cost for this test, this influence may effect the hadoop connection of HDFS. Because this is one machine test, we should not have communication cost for this test,I did not record the communication cost.

	Range Query	Range Join	Distance Query	DistanceJoin Query	Hotzone	Hotcell
Runtime(s)	9.67	37.79	9.72	33.01	21.38	148.43
Memory Usage(MB)	498.62	578.71	493.37	582.04	606.76	1031.51

Two Machine(one master and one worker):

	Range Query	Range Join	Distance Query	DistanceJoin Query	Hotzone	Hotcell
Runtime(s)	9.77	39.29	9.72	31.08	21.14	146.24
Memory Usage(MB)	500	577.63	459.62	591.8	605.24	1128.16
Communication cost(KB)-In	14.59	61.76	36.85	30.37	14.35	64.23
Communication cost(KB)-Out	14.12	35.73	20.44	28.58	24.49	95.95

Three Machines(one master and two workers):

	Range Query	Range Join	Distance Query	DistanceJoin Query	Hotzone	Hotcell
Runtime	9.68	37.69	9.63	31.63	21.41	143.78
Memory Usage	501.89	576.40	500.86	577.39	609.95	1073.69
Communication cost(KB)-In	13.67	23.61	5.89	19.29	19.83	184.96
Communication cost(KB)-Out	11.14	33.53	8.74	26.62	33.56	162.53

Based on the result we get, we can analysis them from inner and outer perspectives.

Inner finding: for each test, we find that the range join query runtime and memory usage are more than rang query. The distance join query runtime and memory usage are also cost more than distance query. I think the reason is the different operation. Range query and distance query are using equal operation, but join operator will cost more than equal, thus the runtime and memory usage will be more.Furthermore, we find the hotcell analysis program cost more runtime, memory usage and communication than the hotzone. The reason might be the complicate of queries in the hot cell analysis, I remember we use the multiple query statements include inner join, outer join.... which should cost more than the simple equal operation in hotzone

analysis.

Outer finding: when number of machine increase,the runtime slightly decrease. Because when we share one job with more than one machines, each machine will have less work load. Moreover, when we compare results between two machines and three machines, the range, range join, distance and distance join‘s communication cost reduce when we have more machines, however, the communication cost for hotzone and hotcell increase when numnber of machine increase. We analysis this particular situation and conclude that becasue the first four operations are simple, query work are not big deal. Thus, when number of machine increase, the cost for their communication will be lower. But the implementation for hot cell and hot zone are complicated, we use spark sql statements and multiple join query tasks, so when number of machine increase, the communication cost for hotcell and hotzone increases.

5.2 Different Size of DataSet

We partition the each dataset for operations and functions, We have small, medium, large size of datasets.
Small:¼ of the original dataset
Medium:½ of the original dataset
Large: Whole dataset

When dataset is small:

	Range Query	Range Join	Distance Query	DistanceJoin Query	Hotzone	Hotcell
Runtime(s)	9.27	17.01	9.37	15.53	18.22	128.24
Memory Usage(MB)	493.71	567.49	488.88	566.82	601.86	1057.2

When dataset is medium:

	Range Query	Range Join	Distance Query	DistanceJoin Query	Hotzone	Hotcell
Runtime(s)	9.45	37.59	9.63	31.4	21.29	146.21
Memory Usage(MB)	438.11	573.45	500.93	586.32	609.88	1154.79

When dataset is large:

	Range Query	Range Join	Distance Query	DistanceJoin Query	Hotzone	Hotcell
Runtime(s)	9.68	113.06	9.74	91.49	25.94	186.62
Memory Usage(MB)	500.24	580.06	499.5	583.94	627.17	1095.52

According to theses three tests, we find that when the dataset become larger, simple work like range, range join, distance and distance join‘s run time keep same. However, complicate work like hotcell and hotcell‘s run time increase. We think the reason might be that even the dataset become larger, running time still depend on the complexity of operations, if operations and functions are simple enough, the run time will not increase when the dataset become larger. For memory usage, when the dataset become larger, it does not change very much. We consider that our work is still simpler than the whole system.

6 CONCLUSIONS

In conclusion, we have learn a lot from the experiments, when we notice the changes appear in the system cost, we realize that choose the suitable number of machine and dataset are also important when we work on a job. We should keep the cost as less as possible and try to reduce the running time for every task. We also believe that not only these two parameters we can test on, if we have enough time in the future, we can test on different CPU and cores, these might be other important factors to effect the cost of operations in the system.