Big Data Processing

ECS765P

COURSE WORK

ETHEREUM ANALYSIS

Authored by:

Viswanatha chepuri

MSc FT Big Data Science

Student Number:

CONTENTS

1. **PART A:**
    - Time Analysis of the Transactions recorded in the Dataset.
    - And, Visualization.
2. **PART B:**
    - Top Ten Most Popular Services.
    - Evaluation of Top Ten Smart Contracts by Total Ether received.
3. **PART C:**
    - Top Ten Most Active Miners
4. **PART D: DATA EXPLORATION**
    - Scam Analysis: Popular Scams and its Analysis.
    - Fork the Chain
    - Gas Guzzlers

PART A

Objective:

To analyse the Ethereum transaction dataset, calculate the number of transactions occurring each month, and compute the average transaction value for each month.

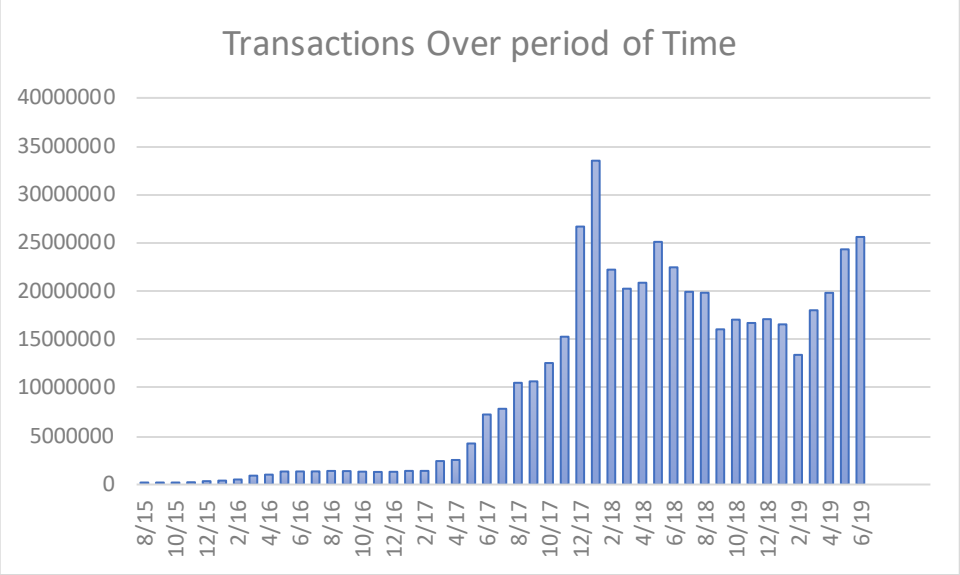
Job ID:
http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_6303/

To determine the number of transactions that have occurred each month over the year, the block_timestamp field in the "Transactions" is used. The aggregation of all values of timestamp per month will fetch the number of transactions that have occurred.

Algorithm:

- Import the required libraries such as mrjob and time to tackle the timestamp, which is in epoch format.
- To use the transactions dataset, as inferred from schema the length of fields is validated.
- Implement a very familiar count operation program.
- Convert the epoch time to Months and Years, and generate it as output for mapper acting as a key.
- The reducer computes the addition of transactions occurring in the month, included a combiner for better operation.
- A text file is with the all data obtained from the above Map-reduce process.
- To improve the fundamental understanding over the data, visualization has been implemented through the software of MS Excel after a few basic pre-processing steps.

The Bar-graph referred in the next page depicted with the horizontal axis as the Month/Year (key) and vertical axis as the number of transactions for that month of that year.



A steady rise in transactions with gaining popularity until end of 2017, with sudden decline that provides itself as evidence to the Great Crypto crash that occurred during that period.

Objective:

To visualize and analyse the data with the help of average value of transaction in each month throughout the data set.

Job ID:

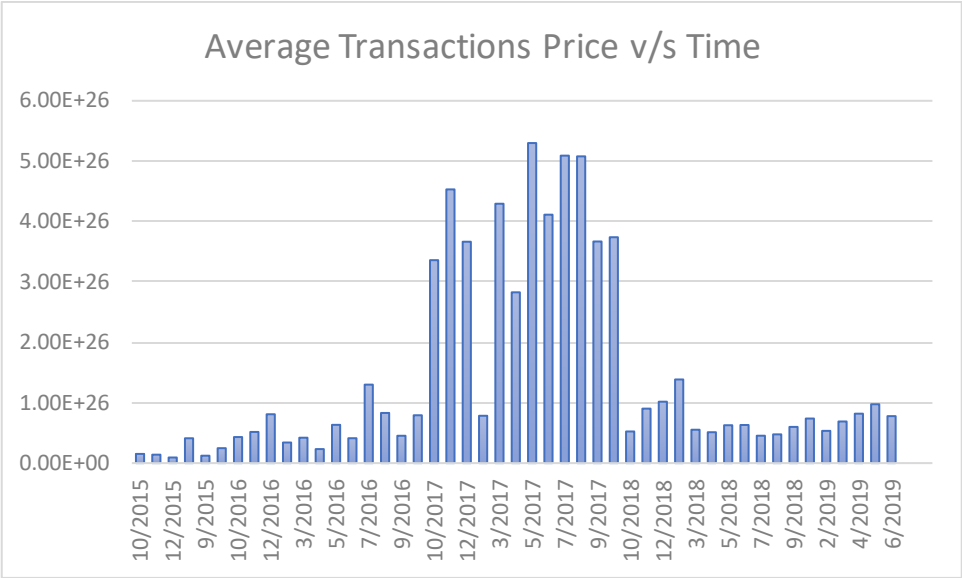
http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_3305/

Similar approach to previous problem. However, in place of number of transactions, the average value(price) of transactions over that month and year is taken into consideration.

Algorithm:

- Import the required libraries such as mrjob and time to tackle the timestamp, which is in epoch format.
- To use the transactions dataset, as inferred from schema the length of fields is validated.
- Implement a very familiar count operation and Average program.
- Convert the epoch time to Months and Years, and generate it as output for mapper acting as a key.
- The reducer computes the addition and average (divides the aggregate by count) of transactions occurring in the month, included a combiner for better operation.

- A text file is with the all data obtained from the above Map-reduce process.
- To improve the fundamental understanding over the data, visualization has been implemented through the software of MS Excel after a few basic pre-processing steps.



After the great crypto-crash a sudden dip in average value of Ether over the months is observed.

PART B

EVALUATE TOP TEN POPULAR SERVICES

SMART CONTRACTS

Job ID:

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_6074/

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_6167/

To determine the top ten most popular services, the addresses of transactions which are present in the contracts depicted as a service are taken as key and aggregated over their transaction values. Hence, two datasets are utilized for running this Map-Reduce.

Since, both datasets have different schema. They can be easily distinguished with fields length.

Algorithm:

- Import the required libraries such as mrjob and MRStep for multiple mappers and reducers.
- To use the transactions dataset, as inferred from schema the length of fields which is seven is validated. With, if condition and the contracts dataset can be for fields length of five inferred from the schema provided.
- Implement a very familiar count operation and Average program.
- The Addresses of both Transactions and Contracts are processed as a key and value as a key value pair from the transaction dataset generating a value of one.
- This value helps in distinguishing between the datasets it from the other dataset. For contracts dataset generated 2(two) as another value and also passing 1(one) which is the counter for the reduce stage.
- Then the reducer checks the values from the values passed from the mapper and if they are true validating the record and its existence.
- Aggregating values and key being the identical passes this computed value as a “value”.
- In the next stage of mapping is the mapper takes the values from the reducer to combine it with the key and values as the it has to evaluate the top ten contracts with None being the key.
- reducer can sort the values in decreasing order implementing a lambda variable function and later iterating through it for obtaining the top ten values and yield.

The obtained output can be seen below, with the list of top ten popular services with address and there aggregated values.

"0xaa1a6e3e6ef20068f7f8d8c835d2d22fd5116444"	84155100809965865822726776
"0xfa52274dd61e1643d2205169732f29114bc240b3"	45787484483189352986478805
"0x7727e5113d1d161373623e5f49fd568b4f543a9e"	45620624001350712557268573
"0x209c4784ab1e8183cf58ca33cb740efbf3fc18ef"	43170356092262468919298969
"0x6fc82a5fe25a5cdb58bc74600a40a69c065263f8"	27068921582019542499882877
"0xbfc39b6f805a9e40e77291aff27aee3c96915bdd"	21104195138093660050000000
"0xe94b04a0fed112f3664e45adb2b8915693dd5ff3"	15562398956802112254719409
"0xbb9bc244d798123fde783fcc1c72d3bb8c189413"	11983608729202893846818681
"0xabbb6bebfa05aa13e908eaa492bd7a8343760477"	11706457177940895521770404
"0x341e790174e3a4d35b65fdc067b6b5634a61caea"	83790007519177556240575000

PART-C TOP TEN ACTIVE MINERS

Objective:

To discover the ten of the most active miners among the miners in the blocks dataset, by aggregating the sizes (block size) of the miner through the dataset and retrieving the information on the ten miners with most largest aggregated size among the miners.

Job ID:
http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_6219/

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_6222/

Algorithm:

- Importing libraries required for the Map-Reduce program execution MRJob (mrjob) and for multiple mappers and reducer, MRStep.
- The schema depicts the length of fields as Nine 9. Validating it using the if condition to except irrelevant data entries.
- Using MRStep can implement multiple map-reducers, the first mapper has the miner as the key and the size as the value which needs to be aggregated.
- The reducer computes the aggregation of the value returned from the mapper, this aggregated value will be yield with its miner key.
- Now, the second mapper combines the both key and value (aggregated) as the value for None assigned as key.
- The key (None) and value (aggregated value and miner) are given as input to the reducer. The reducer then executes the sorting of values in descending order.
- Through iteration only the top ten values are generated as output.

The output can be seen below.

```
"0xea674fdde714fd979de3edf0f56aa9716b898ec8"       23989401188
"0x829bd824b016326a401d083b33d092293333a830"       15010222714
"0x5a0b54d5dc17e0aadc383d2db43b0a0d3e029c4c"       13978859941
"0x52bc44d5378309ee2abf1539bf71de1b7d7be3b5"       10998145387
"0xb2930b35844a230f00e51431acae96fe543a0347"       7842595276
"0x2a65aca4d5fc5b5c859090a6c34d164135398226"       3628875680
"0x4bb96091ee9d802ed039c4d1a5f6216f90f81b01"       1221833144
"0xf3b9d2c81f2b24b0fa0acaaa865b7d9ced5fc2fb"       1152472379
"0x1e9939daaad6924ad004c2560e90804164900341"       1080301927
"0x61c808d82a3ac53231750dadc13c777b59310bd9"       692942577
```

PART D

POPULAR SCAMS

Objective:

To find the most Lucrative form (Type) of Scam. The correlations of this attributes such as status(active/inactive/offline), Volume(value) to its occurrence again and again.

Find the most lucrative form of scam:

JobId:

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_7994/

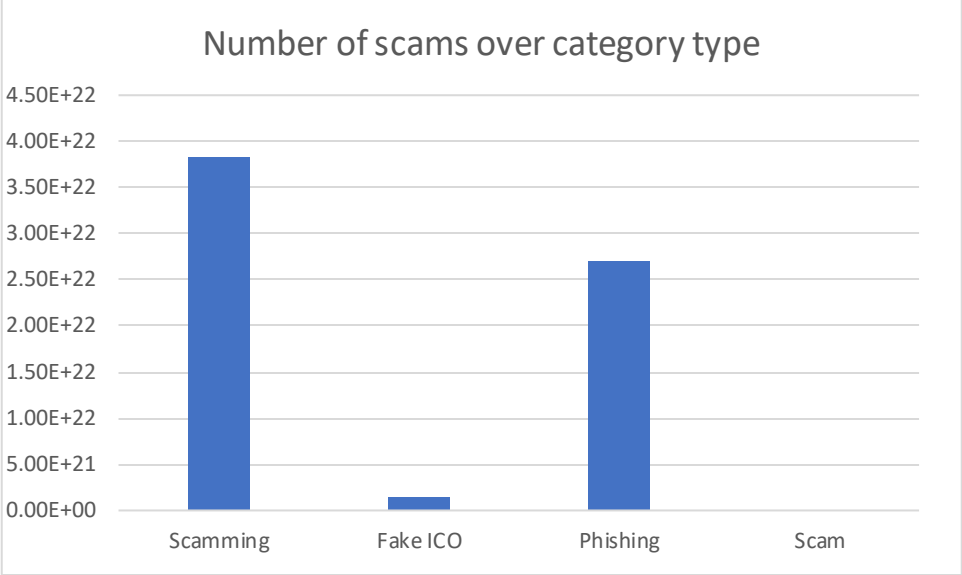
http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_8061/

To execute the scam analysis, the dataset of scams.json in utilized with transactions dataset for the correlation.

Algorithm:

- Importing libraries such as MRJob, MRStep and json.
- Implement if condition to fields validation with the transactions dataset which has seven as the length of fields.
- If the fields does not pass through the condition. It is considered to be scams.json file and will be loaded.
- The address and value (price in Wei) of transactions dataset is taken as the key and value respectively. The value is also assigned with digit zero. Likewise, the addresses in scams.json data obtained through basic iteration,(for loop) as a key and value(category) assigned to one.
- The mapper then yields address and value, 0 for transactions, addresses and category values 0 and 1 from scams dataset with assigned 0 and 1 helping in differentiate the two yields.
- The reducer differentiates the both yields using if condition where values[1] is 0 then the count of values(price) is aggregated. Else, category and transaction value computed sent to next mapper.
- The categories and count of values are key and values respectively for the mapper. With the next reducer computes the aggregation of values over the category acting as key.

The output is visualized for better analysis.



"Scamming"	3.8336162862444 34e+22
"Fake ICO"	1.35645756688963e+21
"Phishing"	2.699937579408742e+22
"Scam"	0

Develop and analyse any relation between scams in category to its status and volume

Job ID:
http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_8089/

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1648683650522_8166/

Algorithm:

- Importing the required libraries MRJob, MRStep, json.
- Use the If condition with length of fields to execute the program correctly for each dataset.
- The addresses in transaction and scams are taken as key for the first mapper assign 0 and 1 similar to previous solution. The scams category and status are taken values.
- The reducer aggregates the values of mapper output of transactions or increments them to categories and status for the scams.json.

- The next mapper inputs the values from the previous reducer, to be sent to next reducer to aggregate the count over the category and status as the key.

The output generated is:

["Active", "Scamming"] 88444

["Inactive", "Phishing"] 22

["Offline", "Fake ICO"] 121

["Offline", "Phishing"] 7022

["Offline", "Scam"] 0

["Suspended", "Phishing"] 11

["Active", "Phishing"] 1584

["Offline", "Scamming"] 24692

["Suspended", "Scamming"] 56

From observing the both output of two jobs, it can be inferred there is larger proportion of scamming type with active status giving larger volume. The scamming and Phishing type with status offline produces the subsequent higher volumes respectively in order.

FORK THE CHAIN

Objective:

To identify and analyse the effects of forks that have occurred in timeline of given dataset.

JobID:

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1649894236110_3582/

Extracted two forks that occurred in the 14th and 15th of December 2017 respectively from the dataset, that are indeed noticed to be adjacent. Hence, can be differentiated and analysed more effectively.

Obtain the related information data on that particular period by executing a MapReduce program.

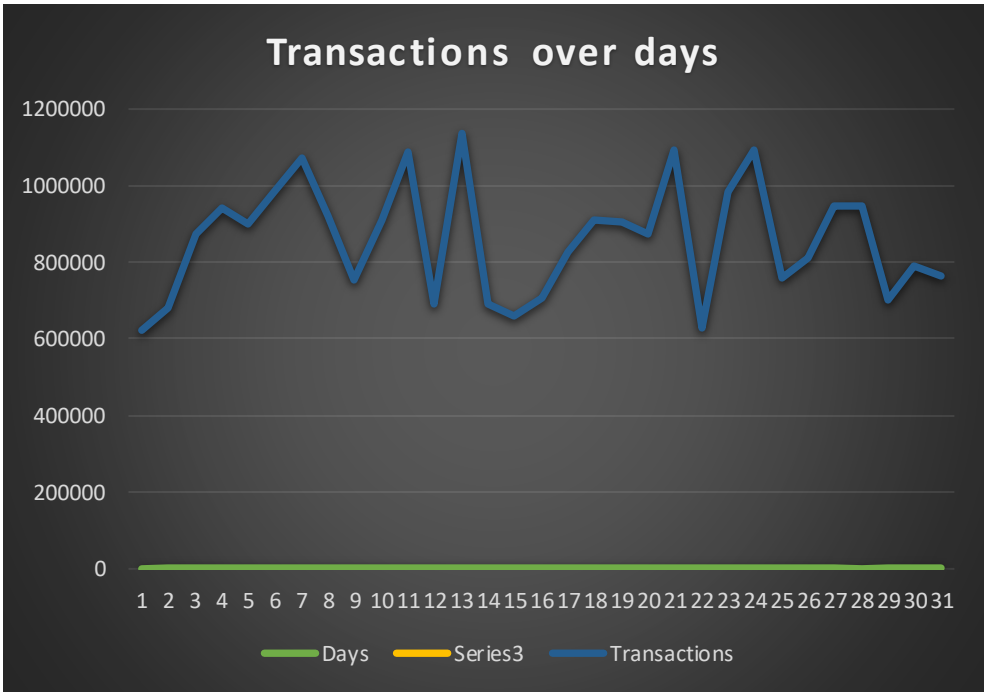
Algorithm:

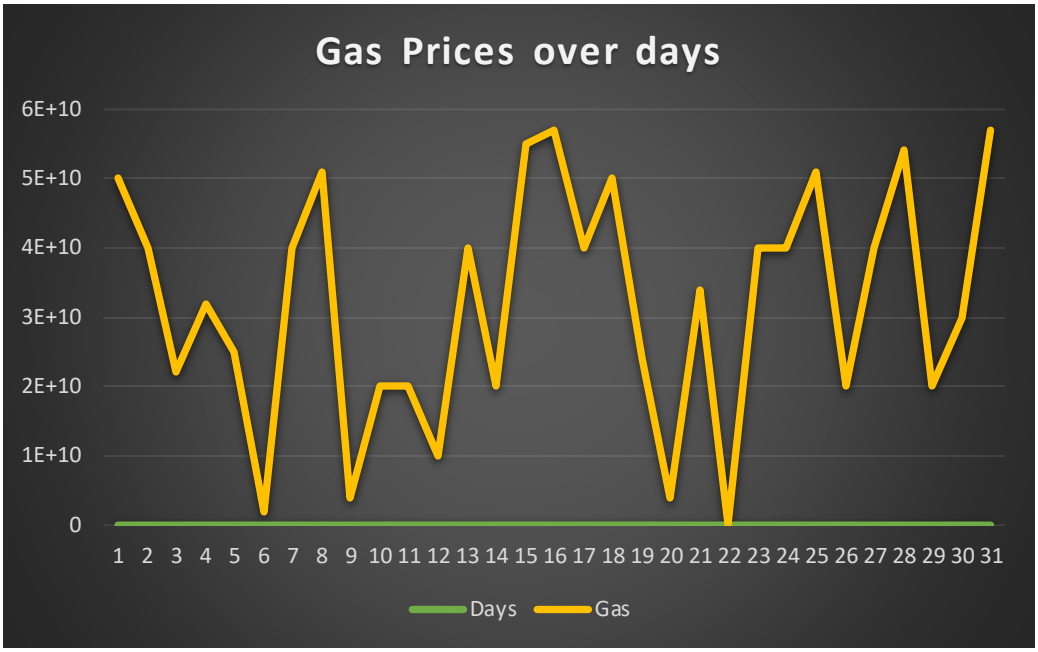
- Importing the libraries such as MRJob and time, as timestamps are to be converted from epoch to our readable format.

- Run over the data and segregate, fields containing each attribute is obtained. With, referring to schema of transactions dataset.
- Yield the readable time format as key and 1(for counting transactions on the day), Gas price as the values.
- The reducer executes the aggregation of count of transactions from values over identical keys and also the value of Gas price over the month for the days.
- Map Reduce executed faster implementing the combiner.
- Yields the readable time format as key and 1(for counting transactions on the day), Gas price as the values from the reducer as the output.

Output obtained is :

1 [622720, 50000000000.0]
10 [681677, 40000000000.0]
31 [946981, 54000000000.0]
.
.
4 [701834, 20000000000.0]
6 [791746, 30000000000.0]
8 [766411, 57000000000.0]





The observation of above two graphs gives overview of effects caused by forking with sudden decline in both transactions and gas subsequently, seen in the two cases. And, also the quick recovery of the same after affected by the fork.

GAS GUZZLERS

Objective:

To perform analysis on the gas parameters affecting the ether and gain insights on the data.

JobId:

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1649894236110_4953/
gas_limit: The maximum gas allowed in this block

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1649894236110_4928/
gas_price : Gas price provided by the sender in Wei

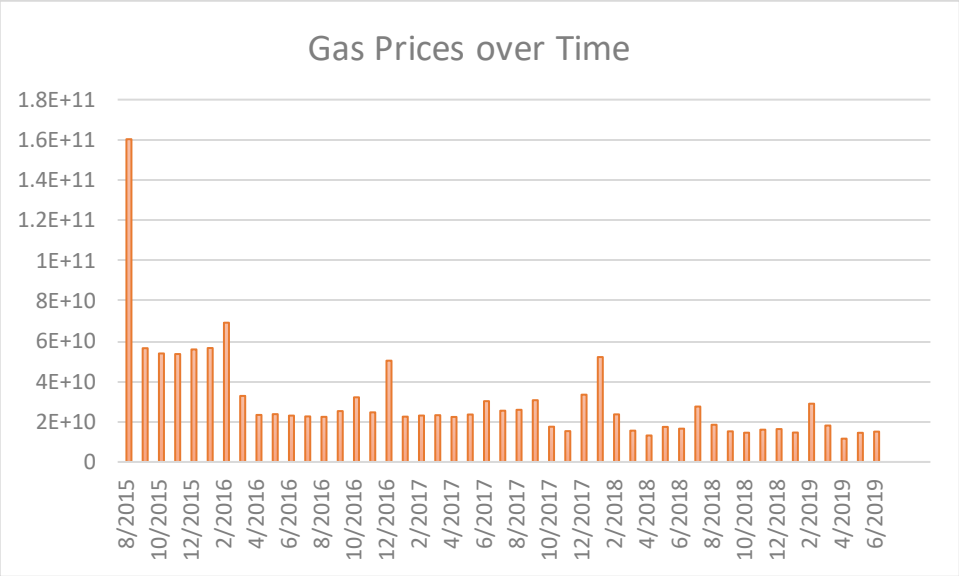
Algorithm:

- Importing the libraries such as MRJob, time.
- Splitting the file lines in accordance to the schema either Transactions or Blocks with data on gas price and gas limit respectively.
- The timestamp is converted to Month and Year format by using functions from the time imported library.
- The mapper yields the key value which are Month & Year (MMYY) and assigned aggregation count value 1(one), either gas limit or gas price respectively.

- The reducer then aggregates the count of transactions under identical key which is Month & Year taken as count. Also, aggregates the values of transactions, blocks values gas prices and gas limit respectively.
- Combiner has implemented for faster execution of MapReduce.
- To summarise the average operation is performed over the key with the values aggregation divided by the maximum count reached. This is given to yield from the reducer as final output.
- The obtained output is visualized using the MS Excel.

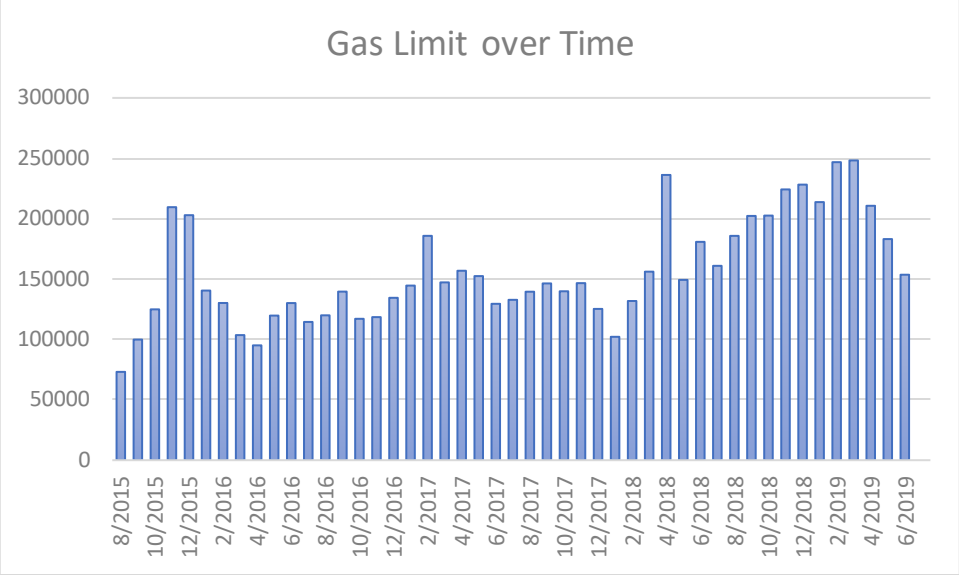
Following output can be expected. For gas limit.

[12, 2017] 125269.51047898784
[2, 2016] 130175.52141758327
[2, 2018] 131824.6970378884
[3, 2017] 147228.19778654756



The gas prices Map reduce will fetch the following output

[10, 2017] 17509171844.770638
[11, 2016] 24634294365.279037
[11, 2018] 16034859008.681646
[12, 2015] 55899526672.35498
[12, 2017] 33423472930.4079
[2, 2016] 69180681134.38954
[2, 2018] 23636574203.82886
[3, 2017] 23232083087.910202



From this it can seen that after rapid sudden decline in gas price at 2015, the gas price has been significantly decreasing while the gas limit has been steadily growing with minor drops and observed at the end of dataset.

The analysis of gas consumption and other factors for the top ten most popular services can be obtained by performing join of output of Part B and the transactions data set taking the gas values.

Job Id:

http://andromeda.student.eecs.qmul.ac.uk:8088/proxy/application_1649894236110_5938/

Algorithm:

- Import the libraries MRJob
- Map the output file of popular services and yield the address as key and assign none with identity number 1 as values.
- The transactions dataset values of gas is extracted and yield as values with addresses being the key.
- The reducer aggregates the total gas value of the service with address being the key and sum(gas prices) as values to yield.