ECS765P
Big Data Processing Ethereum analysis

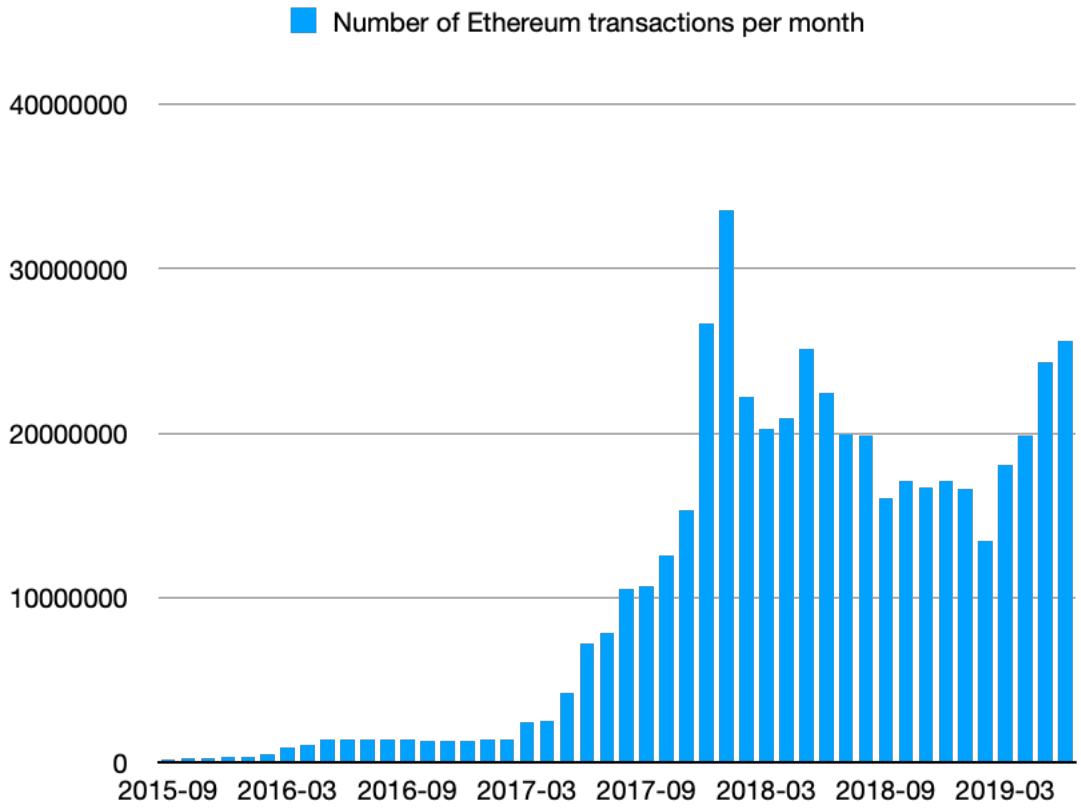
Introduction
This short report consists of four different parts “A, B, C and D” and illustrates them in a respective order. The parts show different types of analysis and implemented with Spark and Hadoop. The analysis is based on Ethereum cryptocurrency and the data shown in this report is from 2015 to 2019. Overall, this project displays the use of Spark and Hadoop with high value data.

Part A

Part 1 – Number of transactions per month

This task implements map reduce, it consists of three parts Mapper, Combiner and Reducer. From every task mapper takes lines of inputs and with the use of *try or except* checks whether it is well formed. The length is 7 due to the dataset is “transactions” and before that there is a split function that splits the line into a list. Then the 6^{th} element is collected “Row time” as int value and it is converted to m_y “month and year” and yields. Combiner then takes all the transactions from the mapper and sums them. Reducer takes the output from the combiner and once again sums transactions.

The following bar chart displays, the transaction started to rapidly increase from early 2017 and reached its’ peak closer to the end of 2017 and beginning of 2018. Then it faced a steady decrease until early 2019, followed by steady increase until the end.



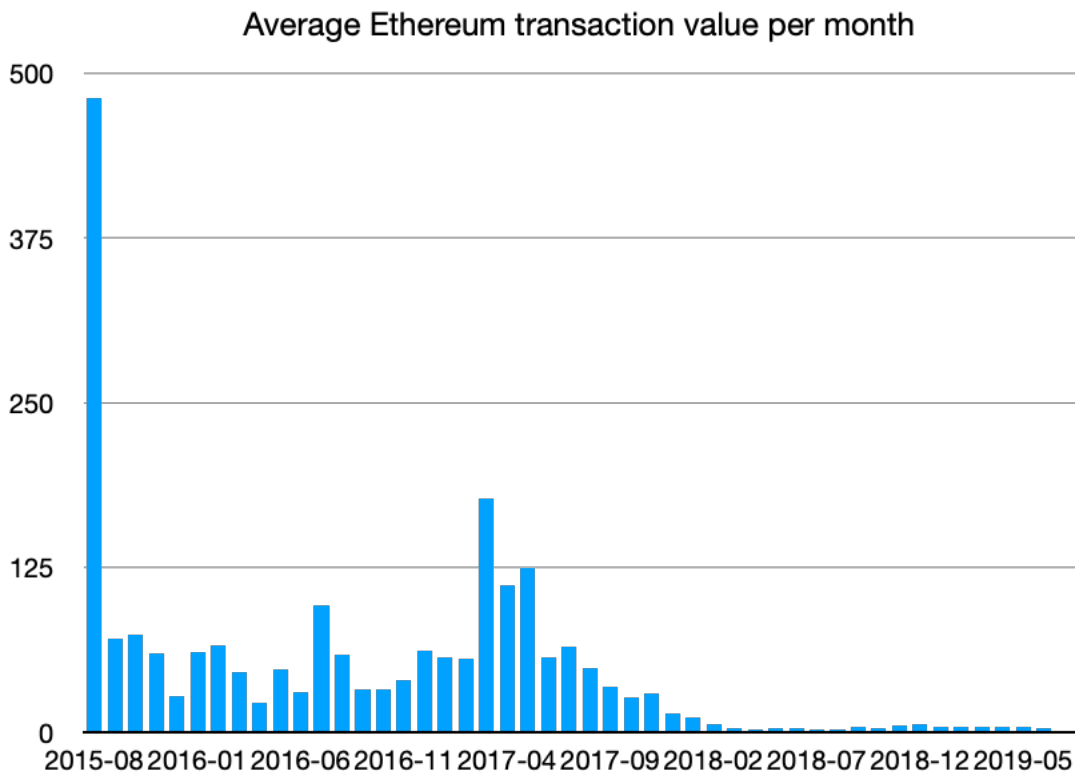

```
job_1648683650522_4263
```

Part 2

Average value of Ethereum transactions per month

This part is similar to the first part but the difference is that the value is collected in float number by dividing 10^{18} to get the Ethereum value. Combiner gets total count and total value and passes to reducer then reducer to get the average value divides total value with total count

The graph shows a dramatic decrease in average transaction value from 500 Ethereum per transaction to somewhere around 100 from June 2015 to January 2015. Keeps fluctuating until late 2016 and comes back with around 170 Ethereum per transaction in 2017. However, faces a decrease till the end and remain at 0 Ethereum per transaction.




```
job_1648683650522_4404
```

Part B

Top ten most popular services.

To complete this task map reduce was used as well as two datasets “Transactions and Contracts” to get the results. From contracts address and block number were used and from transactions to address and value.

According to top ten list above, top 1 has the lion share by having twice more value than others.

Address	Ethereum Value
0xaa1a6e3e6ef20068f7f8d8c835d2d22fd5116444	8.41551E+25
0xfa52274dd61e1643d2205169732f29114bc240b3	4.57875E+25
0x7727e5113d1d161373623e5f49fd568b4f543a9e	4.56206E+25
0x209c4784ab1e8183cf58ca33cb740efbf3fc18ef	4.31704E+25
0x6fc82a5fe25a5cdb58bc74600a40a69c065263f8	2.70689E+25
0xbfc39b6f805a9e40e77291aff27aee3c96915bdd	2.11042E+25
0xe94b04a0fed112f3664e45adb2b8915693dd5ff3	1.55624E+25
0xbb9bc244d798123fde783fcc1c72d3bb8c189413	1.19836E+25
0xabbb6bebfa05aa13e908eaa492bd7a8343760477	1.17065E+25
0x341e790174e3a4d35b65fdc067b6b5634a61caea	8.379E+24

```
job_1648683650522_5833
job_1648683650522_5706
```

Part C

Top ten miners

To complete this task blocks dataset was used similar to top ten transactions but this part only uses one dataset “blocks’ and different categories from the dataset.

Miners	Block size mined
0xea674fdde714fd979de3edf0f56aa9716b898ec8	23989401188
0x829bd824b016326a401d083b33d092293333a830	15010222714
0x5a0b54d5dc17e0aadc383d2db43b0a0d3e029c4c	13978859941
0x52bc44d5378309ee2abf1539bf71de1b7d7be3b5	10998145387
0xb2930b35844a230f00e51431acae96fe543a0347	7842595276
0x2a65aca4d5fc5b5c859090a6c34d164135398226	3628875680
0x4bb96091ee9d802ed039c4d1a5f6216f90f81b01	1221833144
0xf3b9d2c81f2b24b0fa0acaaa865b7d9ced5fc2fb	1152472379
0x1e9939daaad6924ad004c2560e90804164900341	1080301927
0x61c808d82a3ac53231750dadc13c777b59310bd9	692942577

```
job_1648683650522_5938
```

Part D

Part 1 Hadoop and spark comparison analysis

To make this comparison the task from Part B was executed three times both with Spark and Hadoop to see how long does it take to do the same job. Considerable amount difference noticed and the following table illustrates the difference in terms of speed.

Attempt number	Hadoop time in seconds	Spark time in seconds
1	2520	902
2	2408	603
3	2400	608

The results might differ and it depends on how busy the cluster is. According to my data Spark is considerably faster compare to Hadoop. Moving into details Spark's first attempt took approximately 5 more minutes compare to other two attempts. Whereas Hadoop consistently displayed similar timings. In terms of coding, Spark in my opinion is easier to understand and has less coding involved compare to Hadoop.

Hadoop ID

```
job_1649894236110_1390
job_1649894236110_1269
job_1649894236110_1630
```

Spark ID

```
job_1649894236110_1689
job_1649894236110_1779
job_1649894236110_1753
```

Part 2 Scam Analysis

With the help of dataset scams, which has data about both transaction dataset and scam schemes for Ethereum, most popular scam types can be conducted and one of the main reasons of this analysis is to find most lucrative scams with status. In addition to the dataset, python script was used through jupyter notebook to convert JSON file to CSV. This part is implemented with spark.

```
import json
import pandas as pd

data = open('scams.json')
scams = json.load(data)

df = pd.DataFrame(columns=['Address', 'Type of Scam', 'Status'])
keys = scams["result"]
for i in keys:
  record = scams["result"][i]
  category = record["category"]
  addresses = record["addresses"]
  status = record["status"]
  for j in addresses:
    df.loc[len(df)] = [j, category, status]

df.to_csv('scams.csv', index=False)
```

According to the output below phishing takes the lion share with 9526 number of scams, second most popular is Scamming and last one is Fake ICO. Moreover, only two from the list are active and the rest are offline.

```
Phishing, Offline, 3745402749273798820314l, 9526

Scamming, Offline, 22099890651296288506280, 22442

Scamming, Active, 22612205279194883345386, 20001

Phishing, Active, 6256455846464804135847, 3110

Fake ICO, Offline, 1356457566889629979678, 121

Scamming, Suspended, 3710167950000000000, 56

Phishing, Inactive, 14886777707995030337, 22

Phishing, Suspended, 1639908130000000000, 11
```

```
job_1649894236110_1976
```

According to the following table on the other hand Scamming is the most popular followed by Phishing and Fake ICO. Nonetheless, even though the Fake ICO is the least popular but the amount of average volume is significantly higher.

Type of scam	ETH Volume	Number of Scam	Average volume
Scamming	44715.81	42499	1.052161
Phishing	43727.02	12669	3.451497
Fake ICO	1356.46	121	11.210413

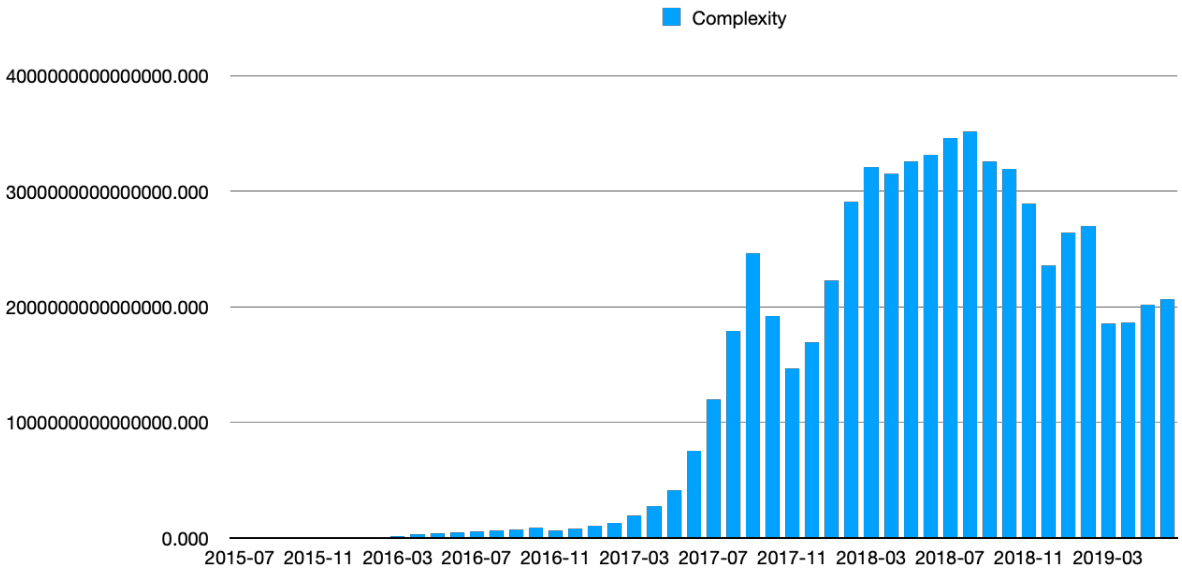
job_1649894236110_1976

Part 3 Gas price Analysis

This section consists of three analysis “Gas price change, Top 10 transaction correlation and complexity analysis”. In order to implement top ten service correlation, I used the address from the output file of top ten transactions. All the analysis are implemented with Hadoop.

Complexity Analysis.

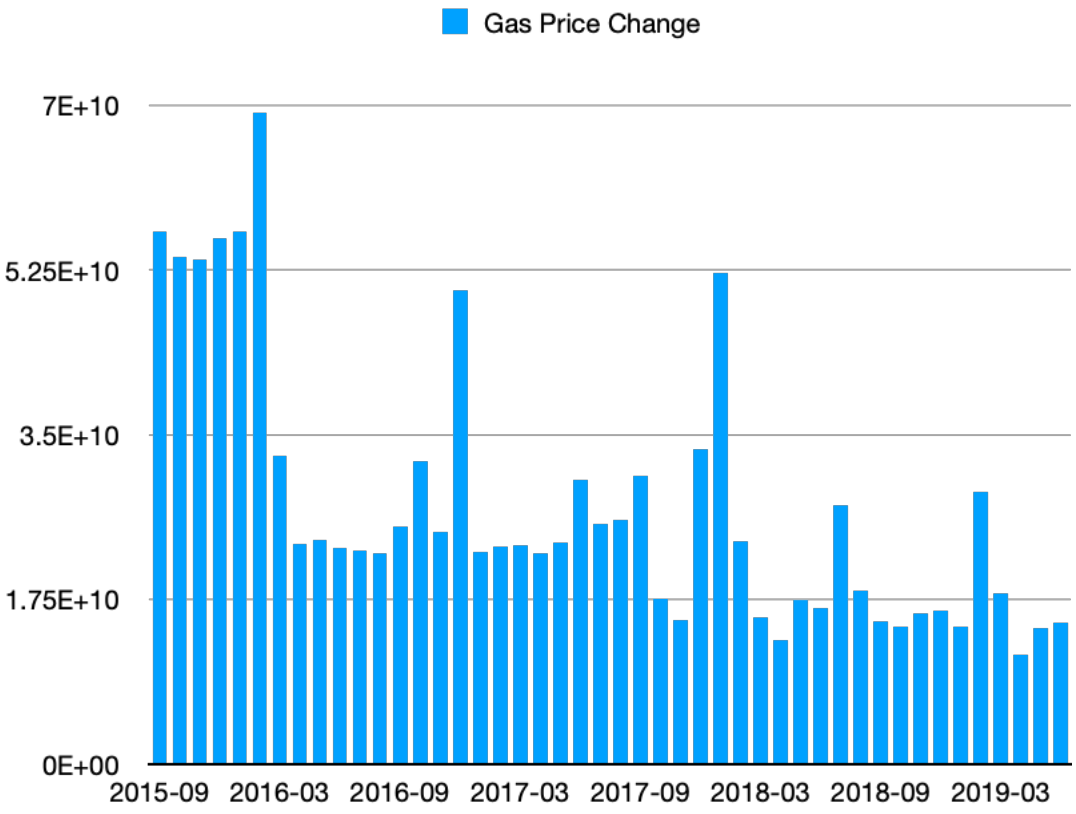
The following graph displays complexity analysis and the peak was 2018-07 and was close to 0 during the beginning from 2015-07 to 2016-11. After reaching its’ peak the graph shows steady decrease and slightly increases by the end of the timeframe.



```
job_1649894236110_1976
```

Gas price Change Analysis

The bar chart below shows the gas price change, the peak was in 2016-03 and faced steady decrease with sudden short term ”two month” rapid increases.



Note: I forgot to add job ID to my file after running the code and now I cannot access it from my local machine. If more information needed ‘bm005’ this is my username please check all my history from Hadoop page.

Top 10 transaction correlation.

The table below shows the most popular services with regard to Ethereum analysis. Taking everything into consideration smaller the block size and amount of gas more popular it gets.

1946708 ["0xe94b04a0fed112f3664e45adb2b8915693dd5ff3"	**3001**	**575668]**
1428757 ["0xbb9bc244d798123fde783fcc1c72d3bb8c189413"	13824	3711215]
1919996 ["0x341e790174e3a4d35b65fdc067b6b5634a61caea"	3999	734742]
1920419 ["0xaa1a6e3e6ef20068f7f8d8c835d2d22fd5116444"	3475	731981]
1935363 ["0xbfc39b6f805a9e40e77291aff27aee3c96915bdd"	1474	228056]
1966054 ["0xfa52274dd61e1643d2205169732f29114bc240b3"	1708	375125]
1969372 ["0x6fc82a5fe25a5cdb58bc74600a40a69c065263f8"	4783	1101578]
2041128 ["0x7727e5113d1d161373623e5f49fd568b4f543a9e"	1015	139259]
2206259 ["0xabbb6bebfa05aa13e908eaa492bd7a8343760477"	6435	1353738]
2462919 ["0x209c4784ab1e8183cf58ca33cb740efbf3fc18ef"	2310	464781]

Note: I forgot to add job ID to my file after running the code and now I cannot access it from my local machine. If more information needed ‘bm005’ this is my username please check all my history from the Hadoop page.