Customer Behavior Simulations

Team: Summer QC-fun

About the idea

Let’s say hypothetically you are the owner of a small business and you want to seed the market with your product. You’d like to know which influencers to give your product to for *free* in order to maximize sales of the product through their connection to others. You’d also like to minimize how many free products you give out.

Well, it turns out that this problem isn’t an easy one, but has an algorithm known as MAXCUT, which can aid in solving the problem. However, there’s a twist -- we’re going to simulate customer behavior over a period of many days, so it is actually multiple MAXCUT problems! Continue on to read about our approach...

Introduction

The objective is to come up with a simulation for a process of buying a product by ***m*** or more customers out of a total of ***N*** customers and try to predict who will be the next ***m*** that will buy it! Our implementation will use the MAXCUT model update of ***m***-weights at a time. Is it possible to find the unitary transformations between the two stages of the network graph? Use this to decide how to resupply the network with free products. If the unitary is well-behaved we suppose that we can use it to evolve the state without doing the variational optimization for a few consecutive stages. We will need a reasonable customer behavior model to simulate the market. We tried execution of the quantum algorithm both on the *ibmq_16_melbourne* and *ibmq_5_yorktown* devices for actual testing.

Our approach

1. For a total of ***N*** customers, what is the optimal number of customers ***m*** who will receive free samples, so that the total number of customers who buy the product will be maximized.
2. Samples are limited for those who already received them and if at some point a certain threshold is reached, then the sample count is reset for everyone.
3. Based on the result of (1), predict who would be the next group of customers who will buy the product based on the relationships between the customers.
4. In turn, it can be predicted how many iterations it will take to reach a given threshold of the total number of customers who will buy the product.
5. Given the anticipated market behavior a reasonable pricing could be planned.

Our approach (cont'd)

6. The ***N*** customers can be modeled with a weighted graph with ***N*** nodes with edges representing the probability of a purchase by a customer by being connected to a customer who has the product. Determination of ***m*** can be solved with the MAXCUT algorithm.
7. For small model space we use Brute Force and Docplex calculations for tuning the model parameters and subroutines debugging.
8. The MAXCUT problem is mapped to an Ising Hamiltonian that can be solved on with the VQE (Variational Quantum Eigensolver) on a quantum computer. We simulate using the qubit layout of the *ibmq_5_yorktown* and *ibmq_16_melbourne* devices for actual testing.
9. A penalty term is included in the Ising Hamiltonian to reduce the weights from those who already received a free product.
10. The matrix of weights W is updated weekly to reflect the market dynamics.

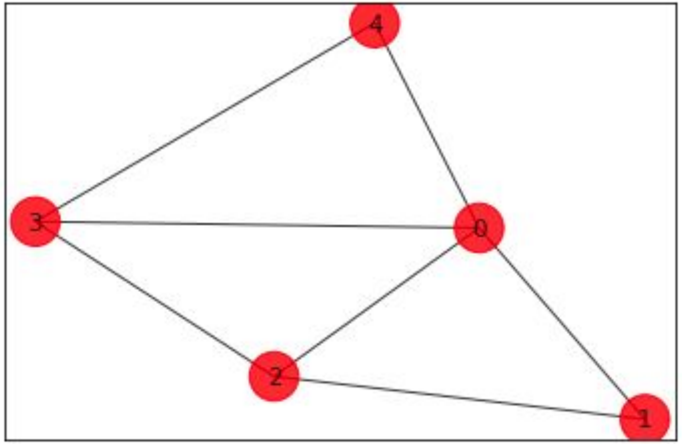
Results

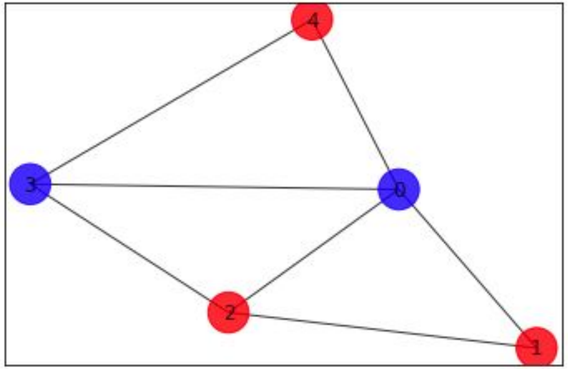
1. Benchmark results from brute-force simulation (5 customers): generally a 2:1 ratio for products purchased to free products given out based on our customer purchase behavior modeling
2. 5 qubit results: reproduced the brute-force result with a 2:1 ratio (both with simulator and *ibmq_5_yorktown*)
3. 15 qubit results: we see poorer results due to qubit connectivity and the type of graph we can represent; these were inconclusive
4. We ran out of time and were not able to define a unitary to model the evolution of purchasing behavior from one simulation day to the next

Example Graph with W-matrix (N=5)

```
Symmetric: True Norm: 3.4017238241954932
Eignvlues: [2.6475469  0.29188844 0.63645966 1.46395822 1.38880054]
[[1.28573075 0.57716683 0.59339038 0.16222185 0.64986048]
 [0.57716683 1.28573075 0.40212449 0.         0.        ]
 [0.59339038 0.40212449 1.28573075 0.56459141 0.        ]
 [0.16222185 0.         0.56459141 1.28573075 0.19689524]
 [0.64986048 0.         0.         0.19689524 1.28573075]]
```

```
Best solution = [1, 0, 0, 1, 0] cost = 2.581904340539446
CPU times: user 40 ms, sys: 8 ms, total: 48 ms
Wall time: 42 ms
```





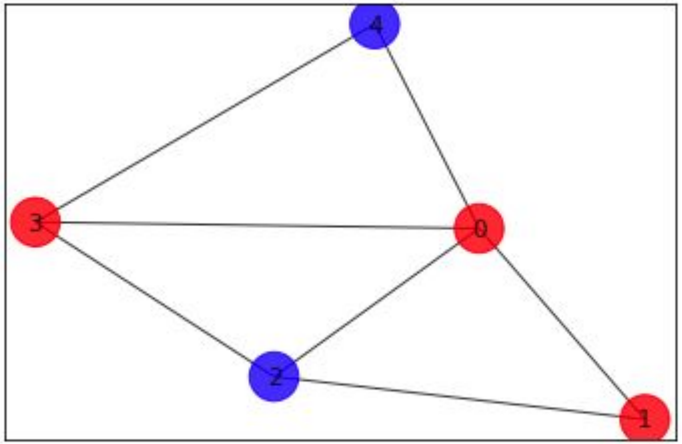
Simulations for 10 days with W-update every 4 days

```
Using Max_Cut_BF
day  [free samples]  [buyers distribution]  [to be used as constrain]
0 [1 0 0 1 0] [0 1 1 0 1] [1 0 0 1 0] 2 3 1.0000
1 [0 0 1 0 1] [1 0 0 1 0] [1 0 1 1 1] 4 5 1.0000
2 [0 0 0 0 0] [1 0 1 0 0] [0 1 0 0 0] 4 7 1.0000
3 [1 0 0 1 0] [0 1 1 0 1] [1 1 0 1 0] 6 10 1.0000
4 [0 0 1 0 1] [1 1 0 0 0] [1 1 1 1 1] 8 12 1.0000
5 [0 0 0 0 0] [1 0 1 1 0] [0 0 0 0 0] 8 15 1.0000
6 [1 0 0 1 0] [0 0 1 0 0] [1 0 0 1 0] 10 16 1.0000
7 [0 0 1 0 1] [1 0 0 1 0] [1 0 1 1 1] 12 18 1.0000
8 [0 0 0 0 0] [1 0 1 0 0] [0 1 0 0 0] 12 20 1.0000
9 [1 0 0 1 0] [0 0 1 0 1] [1 1 0 1 0] 14 22 1.0000
CPU times: user 196 ms, sys: 4 ms, total: 200 ms
Wall time: 206 ms
```

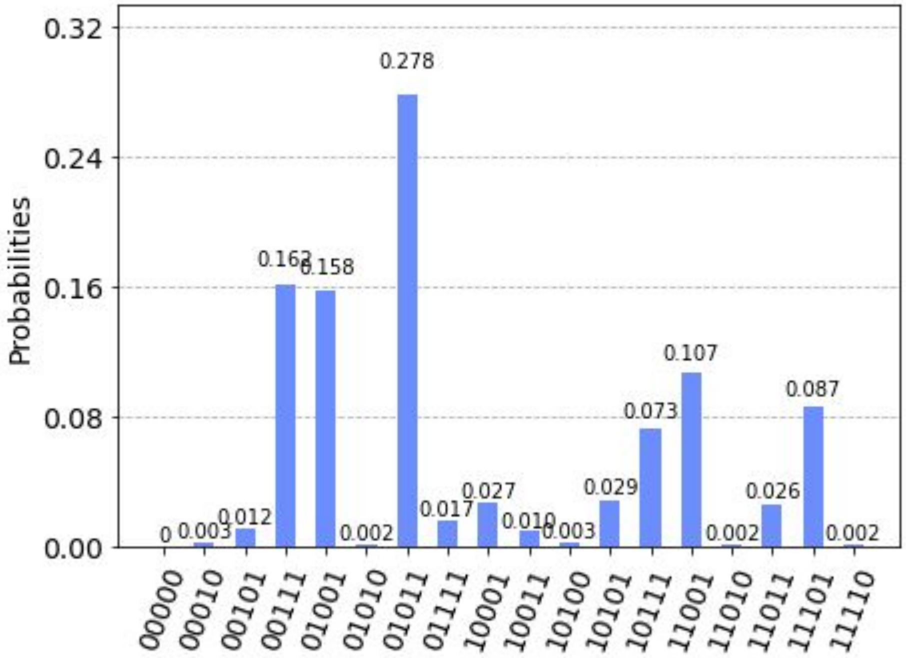
```
Using Max_Cut option: Docplex
day  [free samples]  [buyers distribution]  [to be use
0 [1 0 0 0 0] [0 1 1 0 0] [1 0 0 0 0] 1 2 1.0000
1 [0 0 1 0 1] [1 1 0 0 0] [1 0 1 0 1] 3 4 1.0000
2 [1 0 0 0 0] [0 1 1 0 0] [2 0 1 0 1] 4 6 1.0000
3 [0 0 0 0 0] [0 0 0 0 1] [0 1 0 1 0] 4 7 1.0000
4 [1 0 0 0 0] [0 0 0 0 0] [1 1 0 1 0] 5 7 1.0000
5 [0 0 1 0 1] [1 1 0 0 0] [1 1 1 1 1] 7 9 1.0000
6 [0 0 0 0 0] [0 0 1 0 1] [0 0 0 0 0] 7 11 1.0000
7 [1 0 0 0 0] [0 1 1 0 1] [1 0 0 0 0] 8 14 1.0000
8 [0 0 1 0 1] [1 1 0 1 0] [1 0 1 0 1] 10 17 1.0000
9 [1 0 0 1 0] [0 1 1 0 1] [2 0 1 1 1] 12 20 1.0000
CPU times: user 220 ms, sys: 8 ms, total: 228 ms
Wall time: 4.25 s
```

A run of VQE on the 5-qubit qasam simulator

```
energy: (-0.82340941542414+0j)
time: 2.0448288917541504
max-cut objective: (-2.3965347551200438+0j)
solution: [0 0 1 0 1]
solution objective: 2.406862004554557
```

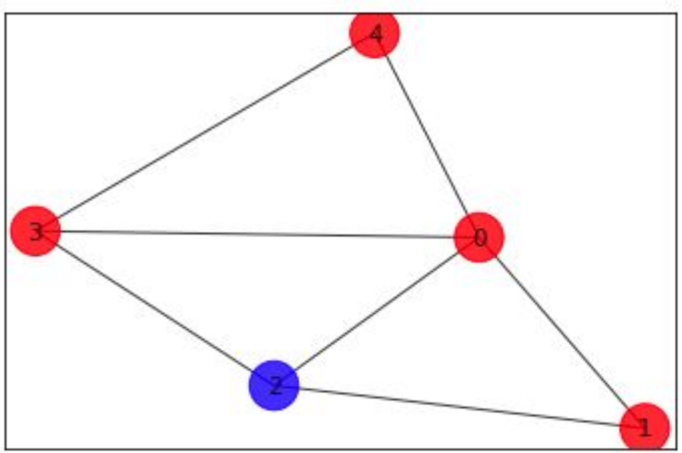


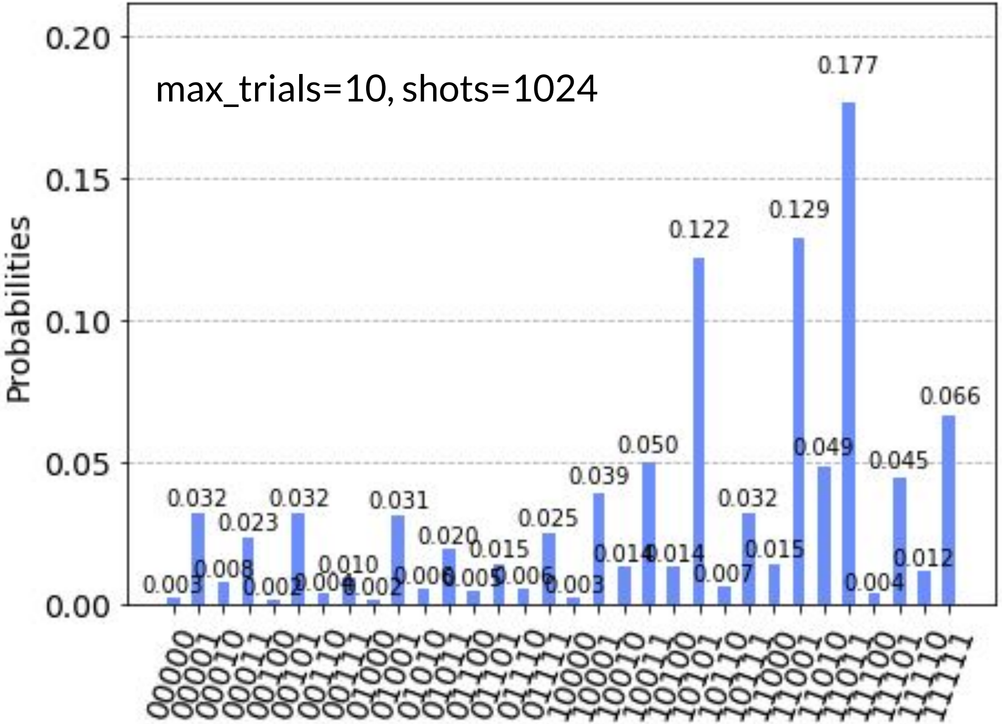
max_trials=30, shots=1024



A run of VQE at 5-qubit device ibmqx2

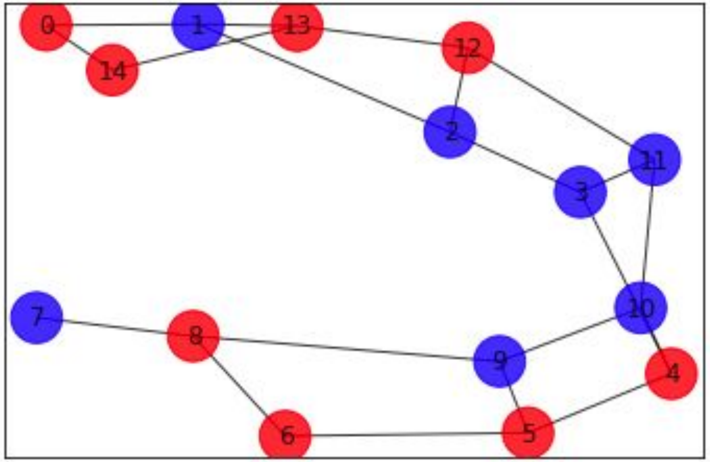
```
energy: (-0.54335454060974+0j)
time: 2043.4817380905151
max-cut objective: (-2.1164798803056435+0j)
solution: [0 0 1 0 0]
solution objective: 1.560106285760894
```

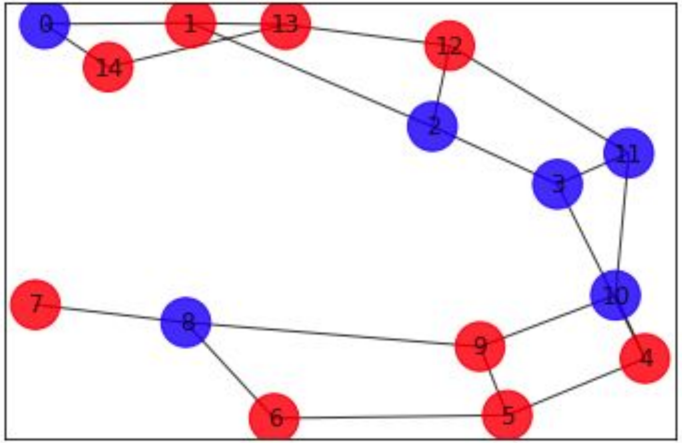




15 q-bits system - ibmq_16_Melbourne

```
energys: [-2.2  -2.2  -2.01]
energy: (-2.1953275349327455+0j) , offset: -2.1953275349327455
max-cut objective: (-4.390655069865491+0j)
solution: [0 1 0 1 0 1 0 0 1 0 1 0 1 0 1]
solution objective: 4.390655069865491
```





```
energys: [-2.63 -2.59 -2.51]
energy: -2.6302834014392342
max-cut objective: 6.424389063628022
solution: [1 0 0 1 0 1 0 0 1 0 1 0 0 1 0]
solution objective: 4.075610936371979
```

N=15 q-bits system - using Qasam & Docplex.

(ibmq_16_Melbourne is still in the queue)

VQE max_trials=10, shots=1024

```
Using Max_Cut option: q
submitting for results using: qasm_simulator
day  [free samples]  [buyers distribution]  [to be used as constraint]  the two totals  w-det
0 [0 0 1 1 0 0 1 1 1 0 1 1 0 1 1] [0 0 0 0 0 0 0 0 0 0 0 0 1 0 0] [0 0 1 1 0 0 1 1 1 0 1 1 0 1 1] 9 1 1.0000
1 [1 0 0 0 0 0 0 1 0 0 0 0 0 1 0] [0 1 0 0 0 0 0 0 1 0 0 0 1 0 0] [1 0 1 1 0 0 1 2 1 0 1 1 0 2 1] 12 4 1.0000
2 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] [1 1 1 1 1 0 0 0 1 0 0 0 0 0 1] [0 1 0 0 1 1 0 0 0 1 0 0 1 0 0] 12 11 1.0000
3 [0 0 0 1 0 0 1 1 0 0 0 1 1 0 1] [1 0 0 0 0 0 0 0 1 0 0 0 0 0 0] [0 1 0 1 1 1 1 1 0 1 0 1 2 0 1] 18 13 1.0000
4 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] [1 0 0 0 0 0 0 0 1 1 0 1 0 0 0] [1 0 1 0 0 0 0 0 1 0 1 0 0 1 0] 18 17 1.0000
5 [1 0 0 0 0 0 1 1 0 0 1 0 0 0 0] [0 0 0 0 0 0 0 0 1 0 0 0 0 0 0] [2 0 1 0 0 0 1 1 1 0 2 0 0 1 0] 22 18 1.0000
6 [0 0 1 0 0 1 0 1 1 1 0 1 0 0 0] [0 0 0 1 0 0 1 0 0 0 0 0 0 0 0] [2 0 2 0 0 1 1 2 2 1 2 1 0 1 0] 28 20 1.0000
7 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] [0 0 0 0 0 0 0 1 0 1 1 0 1 0 0] [0 1 0 1 1 0 0 0 0 0 0 0 1 0 1] 28 24 1.0000
8 [0 1 1 0 0 1 0 1 1 1 1 1 0 1 0] [1 0 0 0 0 0 0 0 0 0 0 0 0 0 1] [0 2 1 1 1 1 0 1 1 1 1 1 1 1 1] 37 26 1.0000
9 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] [1 0 0 1 0 0 0 0 0 0 0 0 0 0 0] [1 0 0 0 0 0 1 0 0 0 0 0 0 0 0] 37 28 1.0000
last day configuration record:
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] [0.39 0.41 0.35 0.4  0.36 0.38 0.3  0.31 0.37 0.41 0.4  0.37 0.37 0.44 0.36]
[1 0 0 1 0 0 0 0 0 0 0 0 0 0 0] 37 28 4.4604 1.0000

CPU times: user 7min 30s, sys: 1.44 s, total: 7min 32s Wall time: 5min 56s
```

```
6 2 1.0000
11 3 1.0000
11 5 1.0000
17 7 1.0000
22 9 1.0000
22 12 1.0000
28 14 1.0000
33 15 1.0000
33 16 1.0000
39 18 1.0000
```

```
Using Max_Cut option: Docplex
[1 0 0 1 0 1 0 0 1 0 1 0 0 1 0]
[-0.76  0.15  0.07 -0.76  0.12 -0.76  0.05  0.07 -0.76  0.17 -0.76  0.1  0.07 -0.76  0.12]
[0 0 0 0 0 0 0 0 0 0 0 0 1 0 1] 39 18 4.4614 1.0000
CPU times: user 1.22 s, sys: 180 ms, total: 1.4 s, Wall time: 4.83 s
```

Code: https://github.com/VGGatGitHub/2020-QISKit-Summer-Jam

Future

1. There is a need for further error analysis, mitigation, and correction.
2. Experiment with other customer purchase behavior models.
3. Explore alternative ways for solving the Ising Hamiltonian problem other than VQE.
4. Publication of the results in appropriate media form.
5. Potential to further develop it as a tool for real world commercial applications.