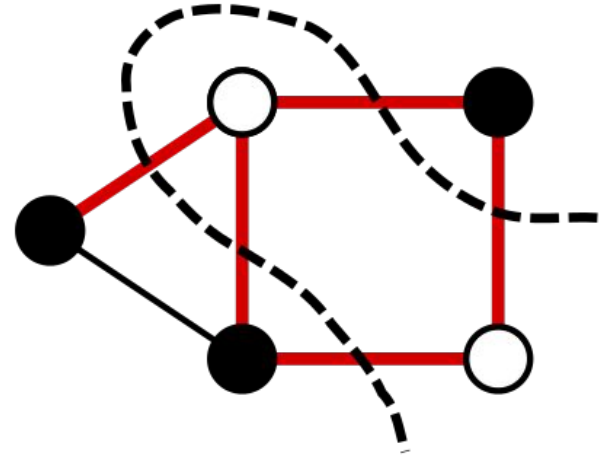


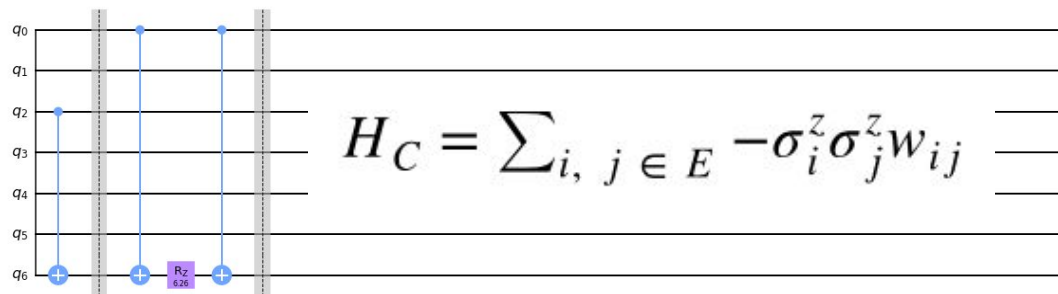
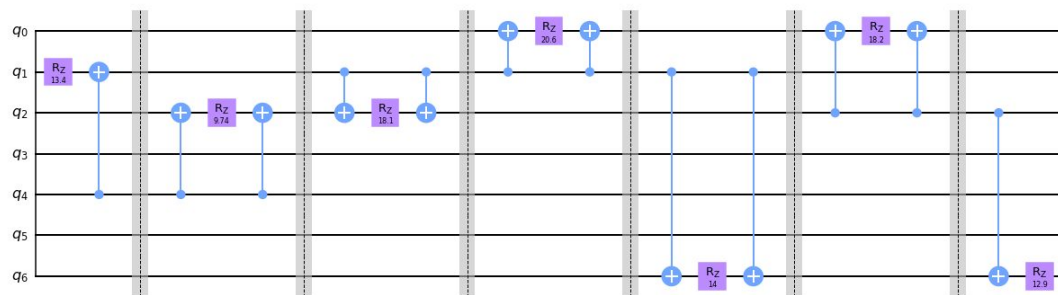
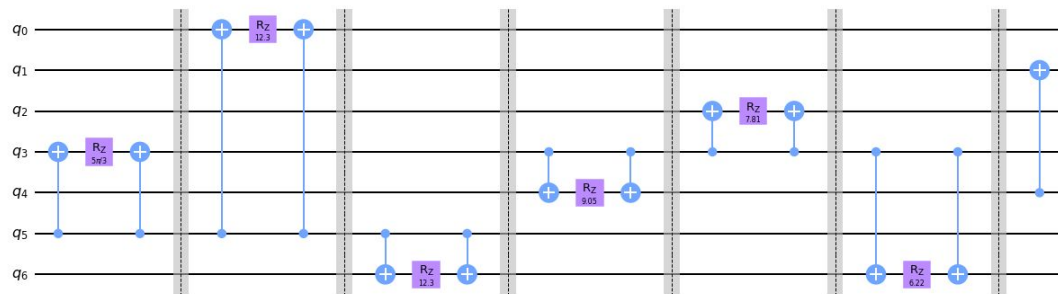
Maxcut Comparisons

Avneesh Verma

Maxcut

- **Objective:** Divide the set of nodes into two subsets, such that the number of edges between nodes of opposite subsets can be maximized.
- In other words, find a way to paint the nodes using a black and white paint brush, and maximize the number of edges between black and white nodes.
- Can be generalized to a weighted case, where each edge is given a certain weight.





$$H_C = \sum_{i, j \in E} -\sigma_i^z \sigma_j^z w_{ij}$$

$$H_B = \sum_{i \in V} \sigma_i^x$$

$$q_0 - \begin{matrix} R_X \\ 2\pi/3 \end{matrix} -$$

$$q_1 - \begin{matrix} R_X \\ 2\pi/3 \end{matrix} -$$

$$q_2 - \begin{matrix} R_X \\ 2\pi/3 \end{matrix} -$$

$$q_3 - \begin{matrix} R_X \\ 2\pi/3 \end{matrix} -$$

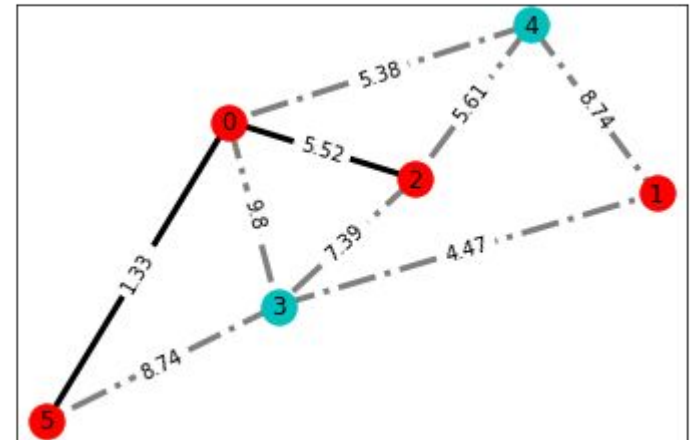
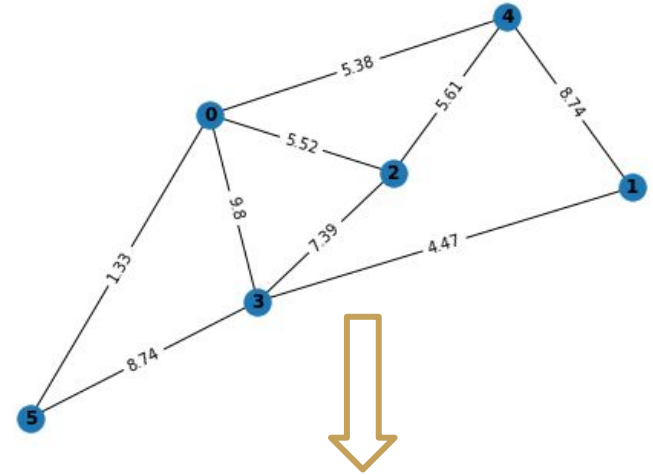
$$q_4 - \begin{matrix} R_X \\ 2\pi/3 \end{matrix} -$$

$$q_5 - \begin{matrix} R_X \\ 2\pi/3 \end{matrix} -$$

$$q_6 - \begin{matrix} R_X \\ 2\pi/3 \end{matrix} -$$

Process

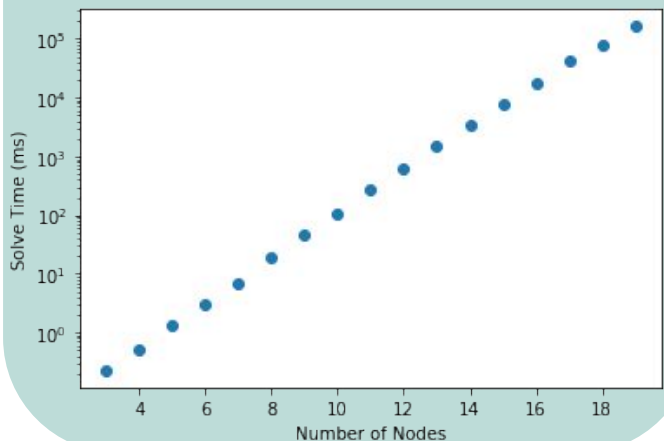
1. Generate an array of graphs, ranging from 3-20 nodes.
2. Use brute-force classical computation to find the exact solution.
3. Build QUBO to solve with D-Wave
4. Generate Circuit to Solve with Qiskit
5. Compare runtimes.



D-WAVE Formulation

```
# Update Q matrix for every edge in the graph
for edge, weight in zip(self.edges, self.weights):
    Q[(edge[0], edge[0])] += -1*weight
    Q[(edge[1], edge[1])] += -1*weight
    Q[(edge[0], edge[1])] += 2*weight
```

Number of Nodes	Solve Time (ms)
3	~2000
4	~2000
5	~2000
6	~2000
7	~2000
8	~2000
9	~2000
10	~2000
11	~2000
12	~2000
13	~2000
14	~5000
15	~10000
16	~20000
17	~45000
18	~78000
19	~160000

A scatter plot showing the relationship between the number of nodes and solve time in microseconds. The x-axis is labeled "Number of Nodes" and ranges from 0 to 18. The y-axis is labeled "Solve Time (microseconds)" and ranges from 28050 to 28350. There are 17 data points plotted as blue dots. The data shows a general upward trend, indicating that solve time increases as the number of nodes increases, though there is some variability in the rate of increase.

Number of Nodes	Solve Time (microseconds)
3	28050
4	28035
5	28115
6	28055
7	28045
8	28160
9	28310
10	28110
11	28195
12	28270
13	28360
14	28180
15	28255
16	28355
17	28120
18	28370
19	28270

1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
1.0
0.9852459663103083
1.0
1.0
1.0
1.0
0.904763022606164

Number of Nodes	Solve Time (s)
3	1.5
4	1.8
5	2.2
6	2.8
7	4.2
8	4.0
9	5.5
10	6.8
11	10.0
12	14.5
13	12.8
14	18.5
15	21.5

```
1.0
0.8345250255362614
0.7145882975906804
0.6200139958012596
0.537231298366294
0.6851159311892296
0.5349421368978201
0.6698457641010357
0.6687413136745425
0.7995365005793741
0.7852047667002351
0.6572806307416142
0.6216935674091045
```