Entropica Labs

OpenQAOA: Bath 2023 Quantum Bootcamp

Contents

OpenQAOA

Your Speaker – Leo

- ➔ @Entropica Labs since 2020
- ➔ MPhys Mathematical Physics @UoE
- ➔ PhD Quantum Information @ParisTech
- ➔ ~3y quantum break (2017-2020)

What is QAOA?

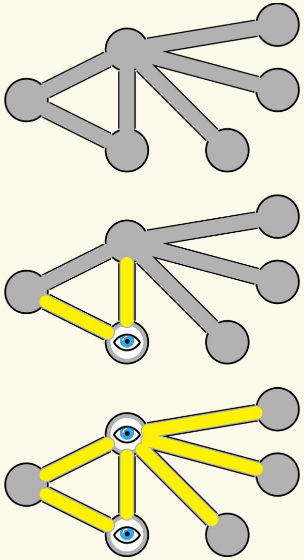
QAOA: Quantum Approximate Optimization Algorithm

'... a quantum algorithm that produces approximate solutions for ***binary*** *combinatorial optimization problems.'*[1]

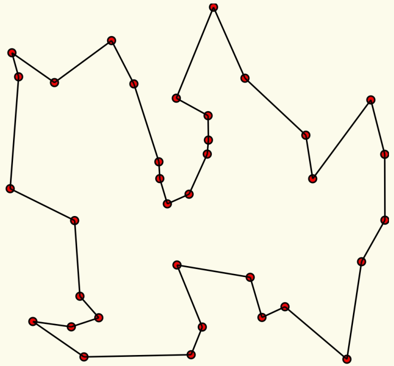
Many binary combinatorial optimization are "NP Complete"

[1] A Quantum Approximate Optimization Algorithm arXiv:1411.4028 [quant-ph]

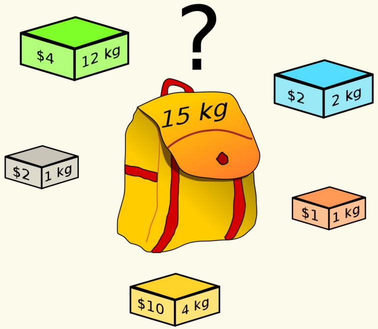
For example



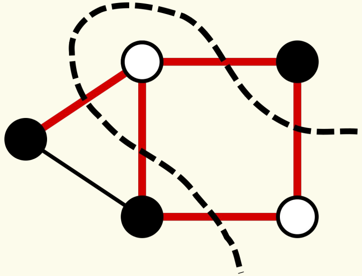
Vertex Cover



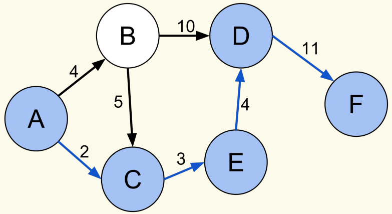
Travelling Salesman



Knapsack Problem

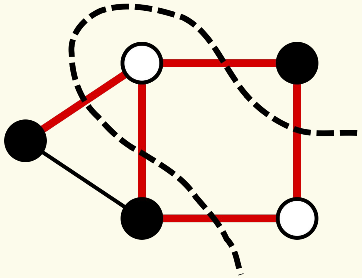


Maxcut



Shortest Path

QAOA



Maxcut

QPU

How do we get started?

1. Define the structure the QAOA
2. Understand how the problem statement is encoded within the QAOA circuit
3. See how to implement it all within OpenQAOA!

Variational Quantum Algorithms

1. Define a **cost function** H_{cost} (it encodes the problem that we want to solve!)

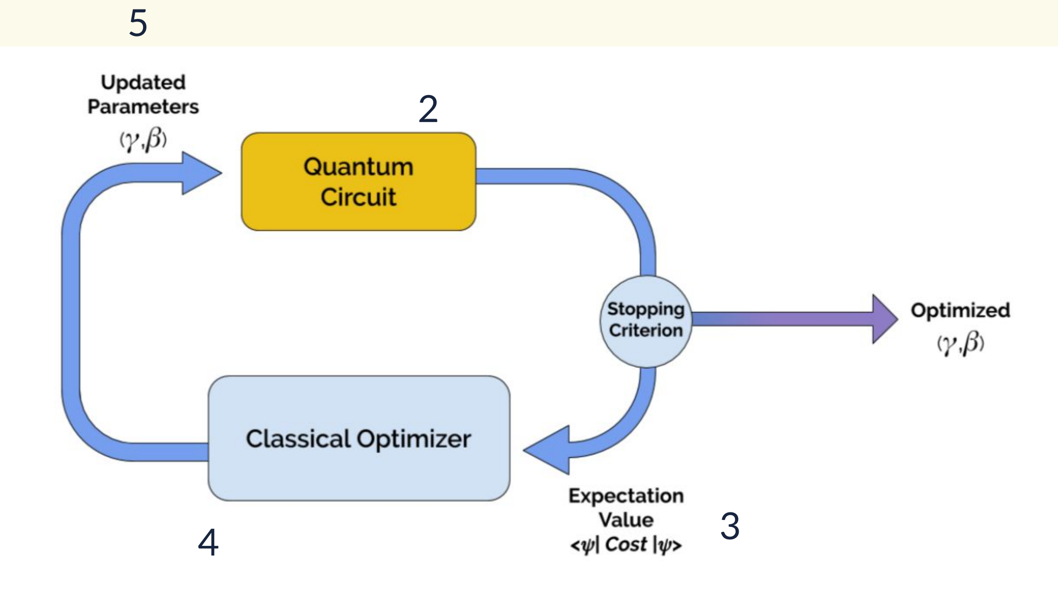
2. Encode H_{cost} in a **Quantum Circuit** parameterized by (β, γ)

3. Measure the expectation value of the cost operator, H_{cost}

4. Use a classical optimiser to propose new parameters (β, γ),

5. Update the quantum circuit accordingly,

6. Repeat until convergence: now have optimal parameters(β*,γ*)



How to find the cost function — QUBOs

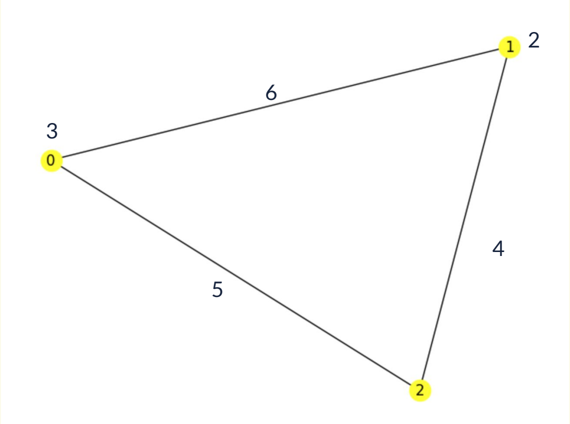
QUBO: Quadratic Unconstrained Binary Optimization

$$C(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=i}^{n} Q_{ij} x_i x_j, \qquad x_i \in \{0, 1\}$$

$x \in \{\pm 1\}$

$$3x_1 + 2x_2 + 6x_1x_2 + 4x_2x_3 + 5x_1x_3,$$

```
terms = [[0], [1], [0,1], [1,2], [0,2]]
weights = [3, 2, 6, 4, 5]
qubo = QUBO(n=3, terms=terms, weights=weights)
```



From QUBOs to Ising Hamiltonians

Hamiltonians ~ Describes how a system can spend its energy

$$y = -5x_1 - 3x_2 - 8x_3 - 6x_4 + 4x_1x_2 + 8x_1x_3 + 2x_2x_3 + 10x_3x_4$$

$$x_i \in \{0, 1\}$$

$$y' = -5x_1^2 - 3x_2^2 - 8x_3^2 - 6x_4^2 + 4x_1x_2 + 8x_1x_3 + 2x_2x_3 + 10x_3x_4$$
$$= \begin{pmatrix} x_4 & x_3 & x_2 & x_1 \end{pmatrix} \begin{pmatrix} -6 & 5 & 0 & 0 \\ 5 & -8 & 1 & 4 \\ 0 & 1 & -3 & 2 \\ 0 & 4 & 2 & -5 \end{pmatrix} \begin{pmatrix} x_4 \\ x_3 \\ x_2 \\ x_1 \end{pmatrix}$$

$$x_i \leftrightarrow \frac{\mathbb{I} - Z_i}{2}$$

$$H_y = -\frac{1}{4}\left[5(\mathbb{I} - Z_1)^2 + 3(\mathbb{I} - Z_2)^2 + 8(\mathbb{I} - Z_3)^2 + 6(\mathbb{I} - Z_4)^2\right]$$
$$+\frac{1}{4}\left[4(\mathbb{I} - Z_1)(\mathbb{I} - Z_2) + 8(\mathbb{I} - Z_1)(\mathbb{I} - Z_3) + 2(\mathbb{I} - Z_2)(\mathbb{I} - Z_3) + 10(\mathbb{I} - Z_3)(\mathbb{I} - Z_4)\right]$$

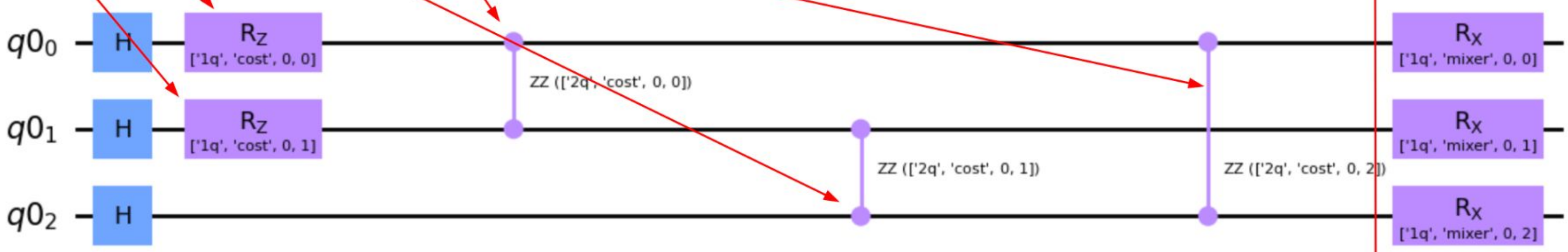
From the Hamiltonian to the Circuit

```
[9]: qubo.hamiltonian.expression
```

[9]: $0 + 2.0Z_1 + 3.0Z_0 + 4.0Z_1Z_2 + 5.0Z_0Z_2 + 6.0Z_0Z_1$

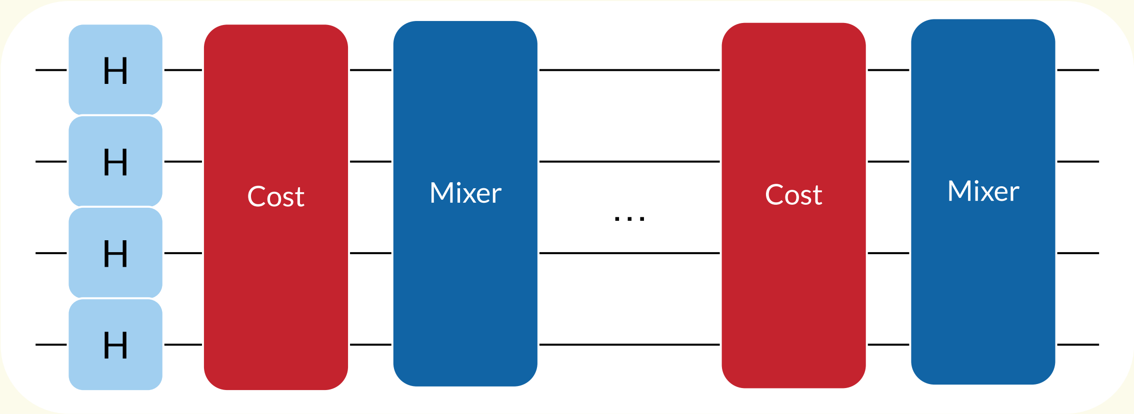
```
[6]: q.backend.parametric_circuit.draw('mpl')
```

[6]:



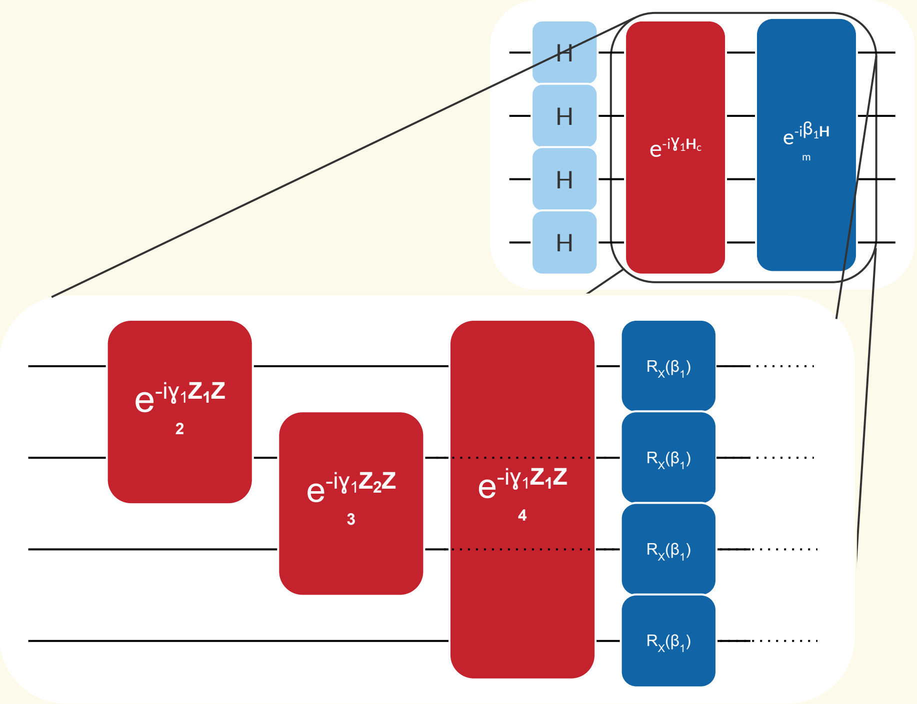
Circuit Structure

https://myqlm.github.io/combinatorial_optimization_intro.html



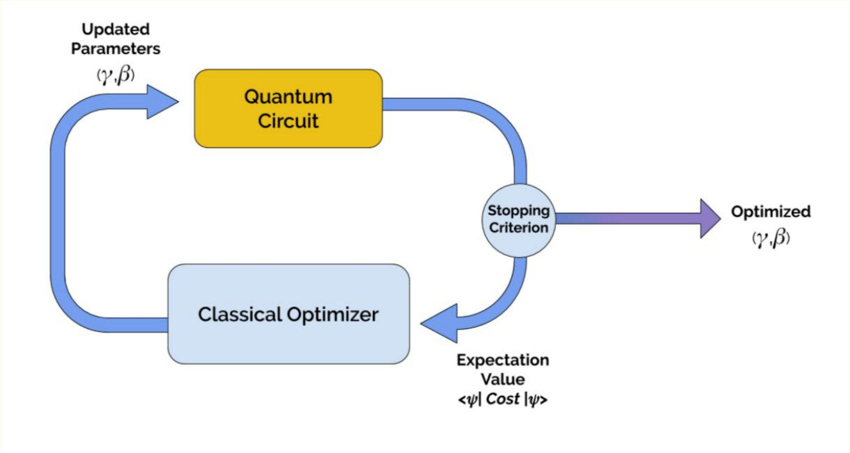
A representational QAOA circuit constructed by alternating application of mixing and cost unitaries.

Circuit Structure

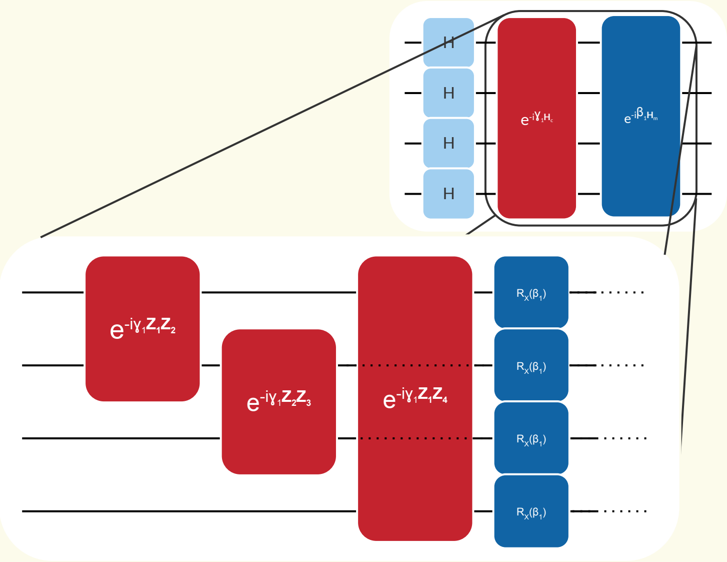


Each alternating block of unitaries can be further broken down into 1 and 2-qubit interactions.

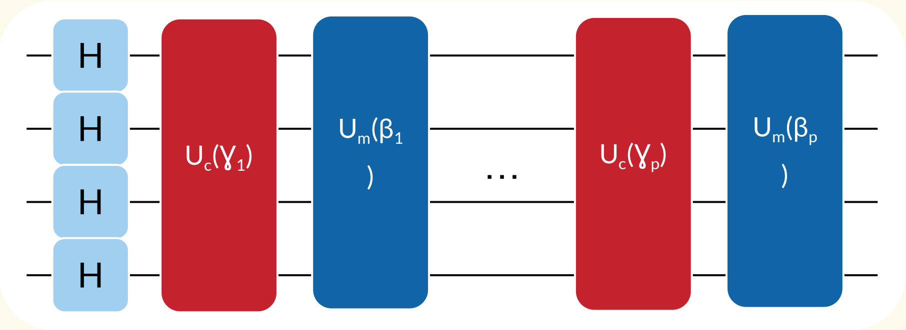
Summary



Updated Parameters
(γ, β)
Quantum Circuit
Stopping Criterion
Optimized
(γ, β)
Classical Optimizer
Expectation Value
<ψ| Cost |ψ>



H
H
H
H
$e^{-i\gamma_1 H_c}$
$e^{-i\beta_1 H_m}$
$e^{-i\gamma_1 Z_1 Z_2}$
$e^{-i\gamma_1 Z_2 Z_3}$
$e^{-i\gamma_1 Z_1 Z_4}$
$R_X(\beta_1)$
$R_X(\beta_1)$
$R_X(\beta_1)$
$R_X(\beta_1)$



H
H
H
H
$U_c(\gamma_1)$
$U_m(\beta_1)$
...
$U_c(\gamma_p)$
$U_m(\beta_p)$

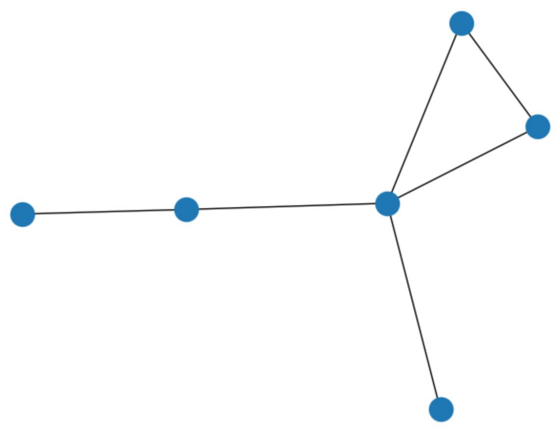
02
An Example with QAOA

Solving MaxCut with OpenQAOA

```
[10]: import openqaoa as oq
      from openqaoa.problems import MaximumCut

[13]: maxcut = MaximumCut.random_instance(n_nodes=6, edge_probability=1)

      # Convert to a qubo!
      maxcut_qubo = maxcut.qubo

[14]: q = oq.QAOA()

[15]: device = oq.create_device(location='local', name='vectorized')
      q.set_device(device)

[16]: q.compile(qubo)

[17]: q.optimize()
```

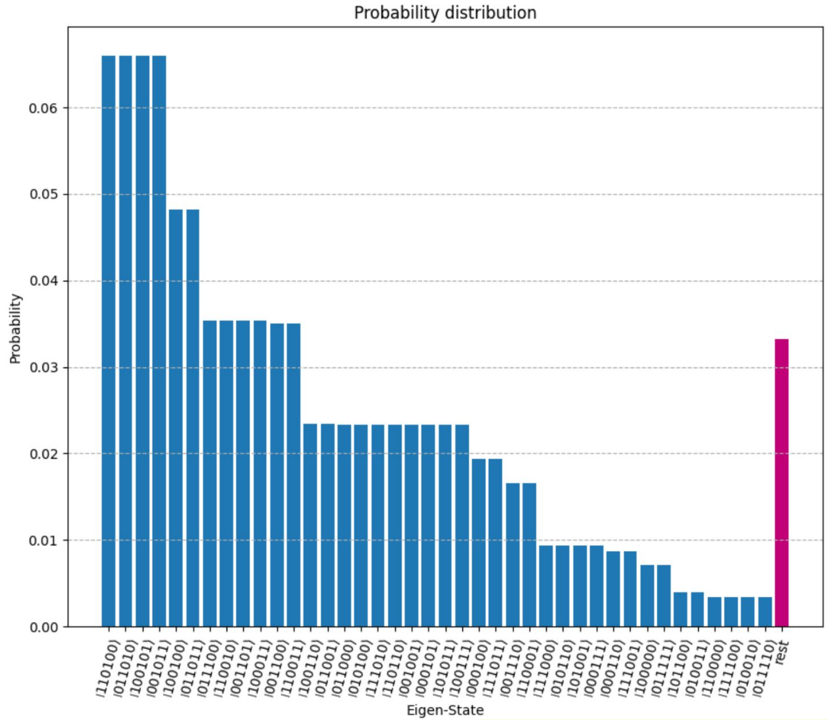
Interpret the result

```python
import networkx as nx
nx.draw(maxcut.G)
```



```python
q.result.plot_probabilities()
```

```
states kept: 40
```

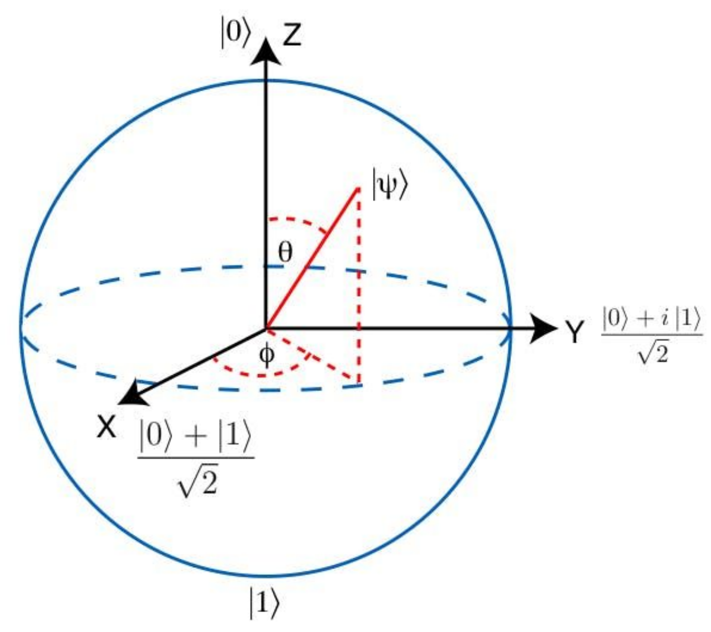



```python
# import the brute-force solver to obtain exact solution
from openqaoa.utilities import ground_state_hamiltonian

energy, configuration = ground_state_hamiltonian(q.cost_hamil)
print(f"Ground State energy: {energy}, Solution: {configuration}")
```

```
Ground State energy: -4.0, Solution: ['100100', '110100', '011010', '100101', '001011', '011011']
```

Wait a second ... {+1,-1} or {0,1} ??

A single qubit!



Z - Measurement

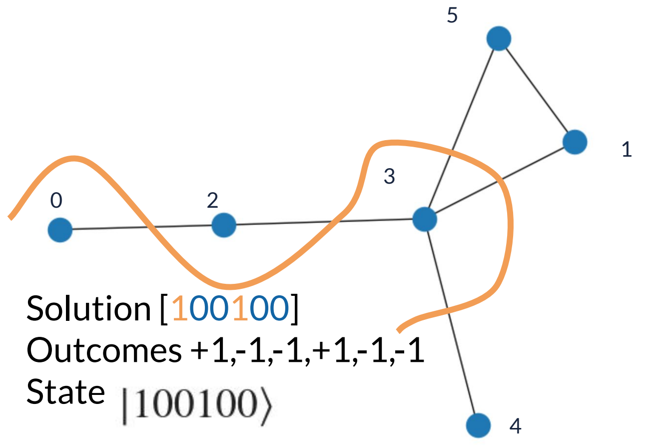
- +1 with probability proportional to $|\psi\rangle$ and $|0\rangle$
- -1 with probability proportional to $|\psi\rangle$ and $|1\rangle$

In other words:

- Observing +1 implies the qubit state is $|0\rangle$
- Observing - 1 implies the qubit state is $|1\rangle$

Interpret the result

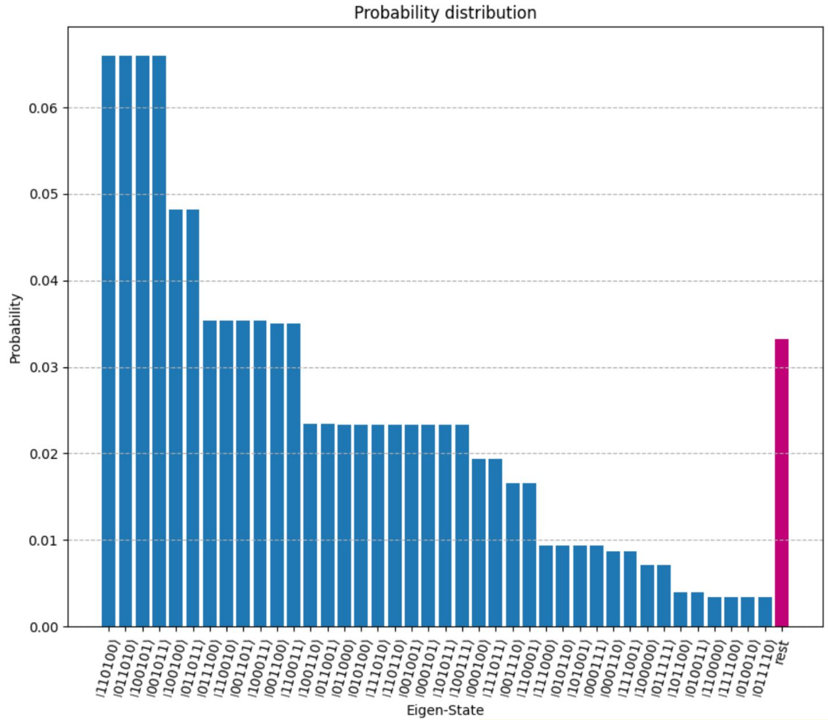
```
import networkx as nx
nx.draw(maxcut.G)
```



Solution [100100]
Outcomes +1,-1,-1,+1,-1,-1
State $|100100\rangle$

```
q.result.plot_probabilities()
```

states kept: 40



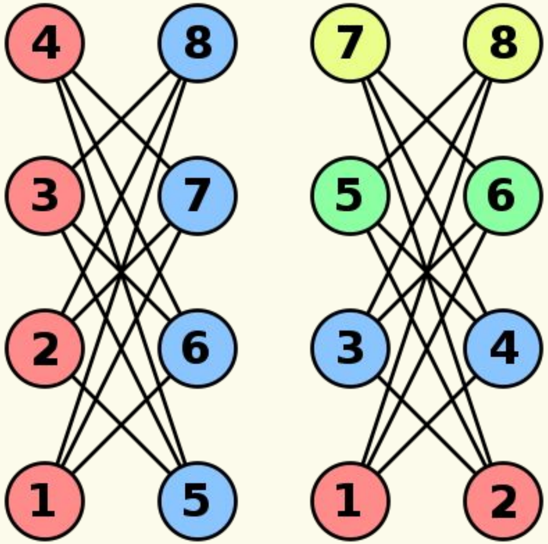

```
# import the brute-force solver to obtain exact solution
from openqaoa.utilities import ground_state_hamiltonian

energy, configuration = ground_state_hamiltonian(q.cost_hamil)
print(f"Ground State energy: {energy}, Solution: {configuration}")
```

Ground State energy: -4.0, Solution: ['100100', '110100', '011010', '100101', '001011', '011011']

03
The Challenge

OpenQA

Solve the graph coloring problem



1. Find the appropriate cost function
2. Convert it to the (ising) {+1, -1} encoding
3. Solve the problem using OpenQAOA
4. Try some extra challenges!

04
Entropica Labs

OpenQAOA

Entropica's Journey

< 2018
Academic positions at Singapore's Centre for Quantum Tech

May 2018
Incorporation of Entropica

Dec 2018
Dev Partners of Rigetti
Members of the Microsoft Quantum Network

Mar 2019
Members of the IBM Quantum Network

Mar 2020
S$2.6M Seed Round

2022
100s of QPU hours running quantum computations on 80 qubits

Jul 2022
Release of OpenQAOA

2023 and onwards

- Sponsor @Qhack 2023
- Strengthen the community behind OQ
- We may open internships in summer 2023
- Loads of low-hanging contributions for OpenQAOA

Thank You.

Per Aspera ad Quantum

To Quantum through Challenges