Quantum algorithms: Noisy intermediate-scale quantum (NISQ), Quantum approximate optimization algorithm

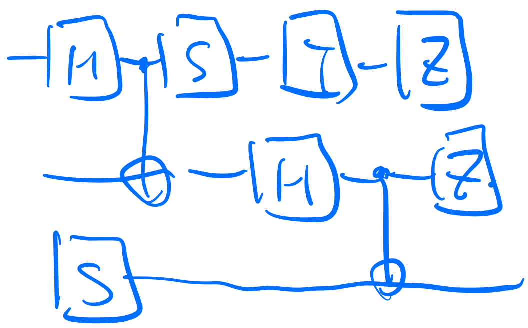
Yipeng Huang

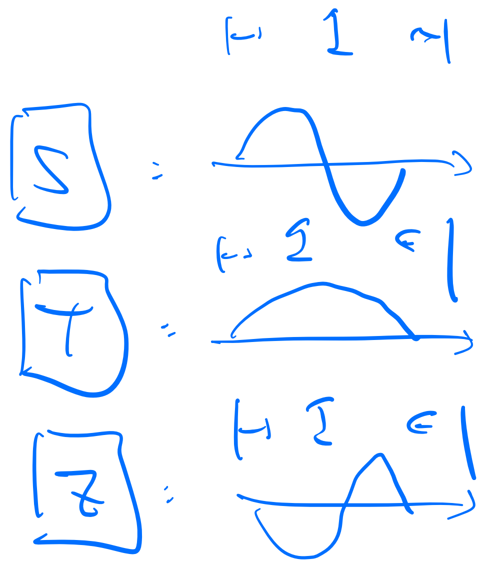
Rutgers University

March 8, 2024

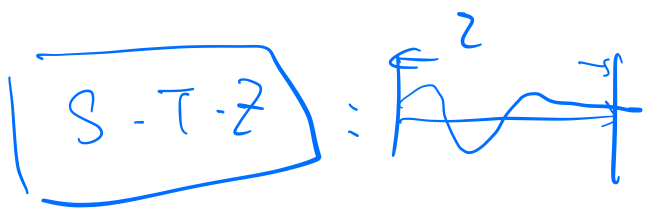
Table of contents

[U] → { Paulis, cnots, T

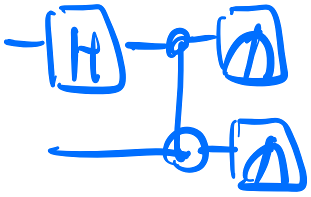




optimal control.



Hardware noise



0. State initial error
1. Decoherence error
2. Gate error (imprecise control of single qubit, two qubit gates)
3. Measurement error

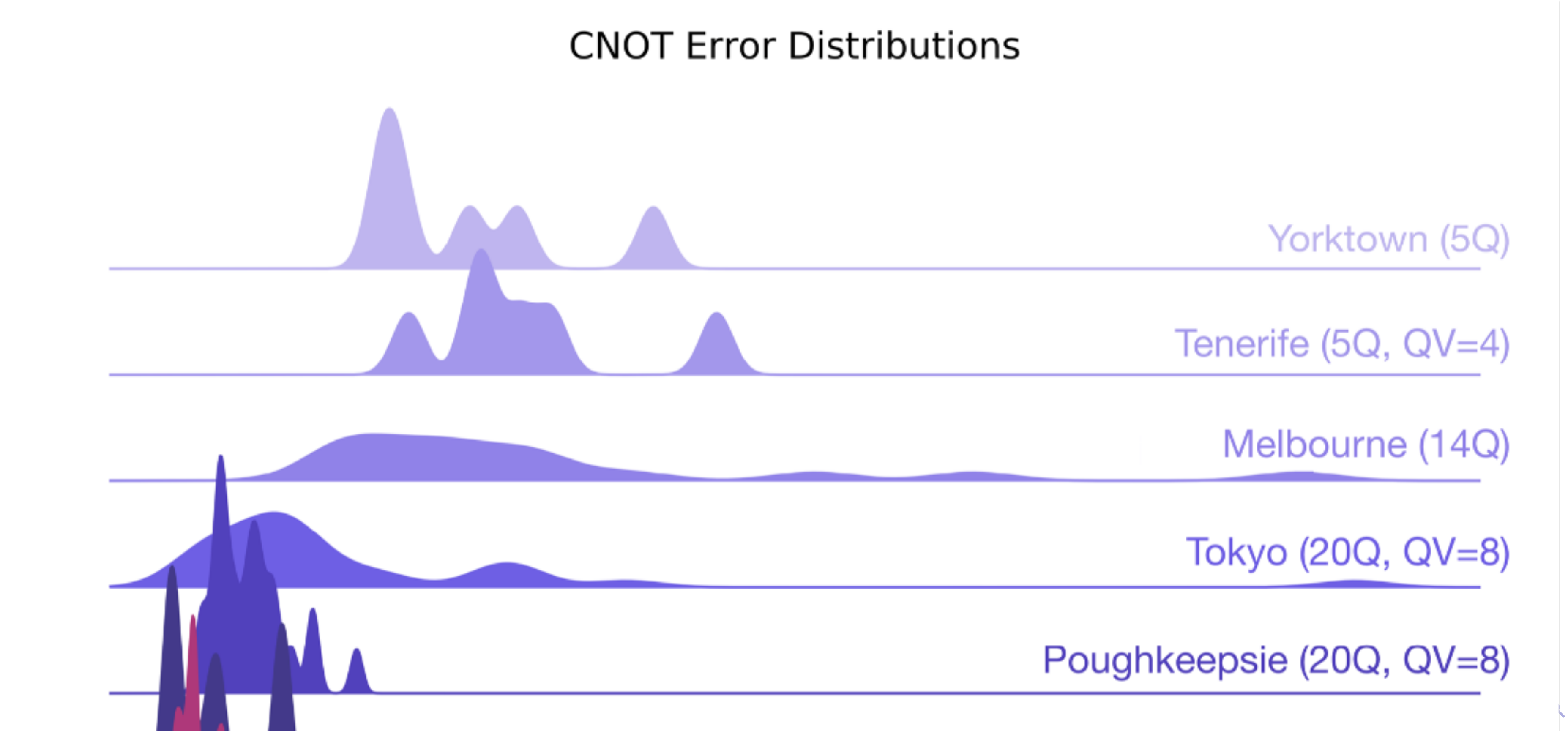
Technology	Coherence Time (s)	1-Qubit Gate Latency (s)	2-Qubit Gate Latency (s)	1-Qubit Gate Fidelity (%)	2-Qubit Gate Fidelity (%)	Mobile
Ion Trap	0.2 [165] - 0.5 [169]	1.6e-6 [166] - 2e-5 [169]	5.4e-7 [166] - 2.5e-4 [169]	99.1 [169] - 99.9999 [168]	97 [169] - 99.9 [165]	YES
Superconductors	7.0e-6 [182] - 9.5e-5 [178]	2.0e-8 [62, 177, 180] - 1.30e-7 [78, 169]	3.0e-8 [182] - 2.5e-7 [78, 169]	98 [179] - 99.92 [177]	96.5 [78, 169] - 99.4 [177]	NO
Solid State Nuclear spin	0.6 [183]	1.12e-4 [184] - 1.5e-4 [183]	1.2e-4 [185]*	99.6 - [184] - 99.95 [183]	89 [186] - 96 [185]*	NO
Solid State Electron spin	1e-3 [3]	3.0e-6 [183] - 2.3e-5 [184]	1.2e-4 [185]*	99.4 [184] - 99.93 [183]	89 [186] - 96 [185]*	NO
Quantum Dot	1e-6 [3, 187] - 4e-4 [173]	1e-9 [3] - 2e-8 [171]	1e-7 [174]	98.6 [171] - 99.9 [172]	90 [171]	NO
NMR	16.7 [158]	2.5e-4 [158] - 1e-3 [24]	2.7e-3 [158] - 1.0e-2 [24]	98.74 [24] - 99.60 [158]	98.23 [24] - 98.77 [158]	NO

Table 1. Metrics for various quantum technologies. * Nuclear/Electron Hybrid

Figure: Credit: [Resch and Karpuzcu, 2019]

Hardware noise

1. Decoherence error
2. Gate error (imprecise control of single qubit, two qubit gates)
3. Measurement error



0. Ideal noise

1. Single parameter depolarizing noise. (uniform)

2. $(\alpha|0\rangle + \beta|1\rangle)$ X flip } (Bit flip)
Y flip (balanced)
Z flip } (phase flip)

3. $|0\rangle \leftrightarrow |1\rangle$ amplitude damping (biased)
phase damping but independent

4. correlated noise

Hardware noise

Stochastic, uncorrelated noise.

	Quantum noise mixtures (Pauli errors)	Quantum noise channels
Pauli-X type	Bit flip noise	Amplitude damping noise (related to T1 time)
Pauli-Z type	Phase flip noise	Phase damping noise (related to T2 time)
Combinations	Symmetric / asymmetric depolarizing noise	Generalized amplitude damping
Simulation technique	Can model as probabilistic ensembles of state vectors	Requires density matrix representation

Table: Summary of canonical quantum noise models.

$$\vec{a} = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta) = (u, v, w)$$

$$\rho = \frac{1}{2}(I + \vec{a}\cdot\vec{\sigma})$$

$$\Delta_\lambda(\rho) = (1-\lambda)\rho + \frac{\lambda}{d} I$$

$$= \sum_{i=0}^{3} K_i \rho K_i^\dagger$$

$$K_0 = \sqrt{1-\frac{3}{4}\lambda}\, I \quad K_1 = \sqrt{\frac{\lambda}{4}}\, X \quad K_2 = \sqrt{\frac{\lambda}{4}}\, Y \quad K_3 = \sqrt{\frac{\lambda}{4}}\, Z$$

$$\sum_i K_i^\dagger K_i = I$$

Bit flip noise channel

$$|0\rangle \to BitFlip(0.64) \to \begin{cases} P(|0\rangle) = 0.64 \\ P(|1\rangle) = 0.36 \end{cases}$$

We represent such a mixture of quantum states as a density matrix:

$$\begin{aligned}
&0.64\,|0\rangle\langle 0| + 0.36\,|1\rangle\langle 1| \\
&= 0.64 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} + 0.36 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} \\
&= 0.64 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + 0.36 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 0.64 & 0 \\ 0 & 0.36 \end{bmatrix}
\end{aligned}$$

(Conventions from [Nielsen and Chuang, 2011, Chapter 8.3])

Density matrix representation

$$0.64\,|0\rangle\langle 0| + 0.36\,|1\rangle\langle 1| = \begin{bmatrix} 0.64 & 0 \\ 0 & 0.36 \end{bmatrix}$$

More general representation:
$\rho = \sum_j p_j \,|\psi_j\rangle\langle\psi_j|$

Quantum (noise) channel

A quantum channel $\mathcal{E}(\rho)$ acts on mixed state ρ:

$\mathcal{E}(\rho) = \sum_k E_k \rho E_k^\dagger$

Bit flip noise channel

The bit flip channel flips the state of a qubit with probability $1 - p$. It has two elements:

$$E_0 = \sqrt{p}I = \sqrt{p}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$E_1 = \sqrt{1-p}X = \sqrt{p}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Bit flip noise channel

The bit flip noise channel $\mathcal{E}_{bitflip}(0.64)$ acts on the $|0\rangle$ state like so:

$$\mathcal{E}_{bitflip}(\begin{bmatrix}1 & 0\\0 & 0\end{bmatrix})$$
$$= \sum_k E_k \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix} E_k^\dagger$$
$$= 0.8\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}0.8\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} + 0.6\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}0.6\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$$
$$= \begin{bmatrix}0.64 & 0\\0 & 0.36\end{bmatrix}$$

Phase flip noise channel

The phase flip channel flips the phase of a qubit with probability $1 - p$. It has two elements:

$$E_0 = \sqrt{p} I = \sqrt{p} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$E_1 = \sqrt{1 - p} Z = \sqrt{p} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Hardware noise

	Quantum noise mixtures (Pauli errors)	Quantum noise channels
Pauli-X type	Bit flip noise	Amplitude damping noise (related to T1 time)
Pauli-Z type	Phase flip noise	Phase damping noise (related to T2 time)
Combinations	Symmetric / asymmetric depolarizing noise	Generalized amplitude damping
Simulation technique	Can model as probabilistic ensembles of state vectors	Requires density matrix representation

Table: Summary of canonical quantum noise models.

Amplitude damping noise channel

The amplitude damping channel leaves $|0\rangle$ alone while probabilistically flipping $|1\rangle$. It has two elements:

$$E_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{bmatrix}$$
$$E_1 = \begin{bmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{bmatrix}$$

γ represents probability that $|1\rangle$ decays to $|0\rangle$

Hardware noise

1. Decoherence error
2. Gate error (imprecise control of single qubit, two qubit gates)
3. Measurement error

Technology	Coherence Time (s)	1-Qubit Gate Latency (s)	2-Qubit Gate Latency (s)	1-Qubit Gate Fidelity (%)	2-Qubit Gate Fidelity (%)	Mobile
Ion Trap	0.2 [165] - 0.5 [169]	1.6e-6 [166] - 2e-5 [169]	5.4e-7 [166] - 2.5e-4 [169]	99.1 [169] - 99.9999 [168]	97 [169] - 99.9 [165]	YES
Superconductors	7.0e-6 [182] - 9.5e-5 [178]	2.0e-8 [62, 177, 180] - 1.30e-7 [78, 169]	3.0e-8 [182] - 2.5e-7 [78, 169]	98 [179] - 99.92 [177]	96.5 [78, 169] - 99.4 [177]	NO
Solid State Nuclear spin	0.6 [183]	1.12e-4 [184] - 1.5e-4 [183]	1.2e-4 [185]*	99.6 - [184] - 99.95 [183]	89 [186] - 96 [185]*	NO
Solid State Electron spin	1e-3 [3]	3.0e-6 [183] - 2.3e-5 [184]	1.2e-4 [185]*	99.4 [184] - 99.93 [183]	89 [186] - 96 [185]*	NO
Quantum Dot	1e-6 [3, 187] - 4e-4 [173]	1e-9 [3] - 2e-8 [171]	1e-7 [174]	98.6 [171] - 99.9 [172]	90 [171]	NO
NMR	16.7 [158]	2.5e-4 [158] - 1e-3 [24]	2.7e-3 [158] - 1.0e-2 [24]	98.74 [24] - 99.60 [158]	98.23 [24] - 98.77 [158]	NO

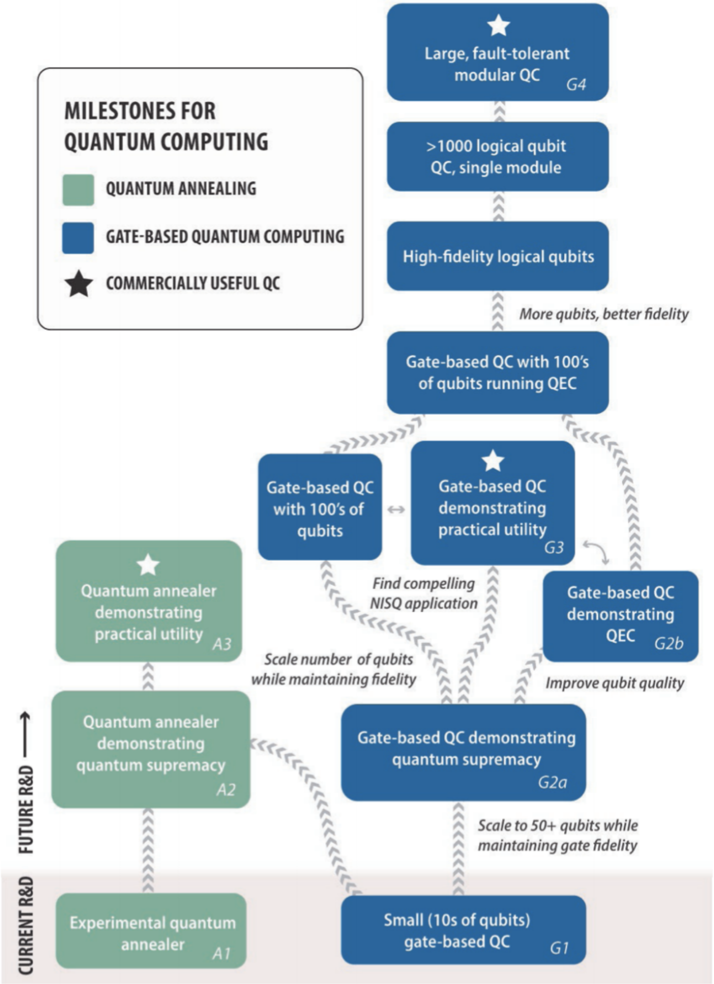
Table 1. Metrics for various quantum technologies. * Nuclear/Electron Hybrid

Figure: Credit: [Resch and Karpuzcu, 2019]

But what about correlated noise events?

Table of contents

Steps from NISQ toward FTQC



An illustration of potential milestones of progress in quan-

- Noisy Intermediate Scale Quantum vs. Fault Tolerant Quantum Computation
- Credit: National Academies of Sciences, Engineering, and Medicine. Quantum Computing: Progress and Prospects. 2019.

Applications of near-term and far-future quantum computing

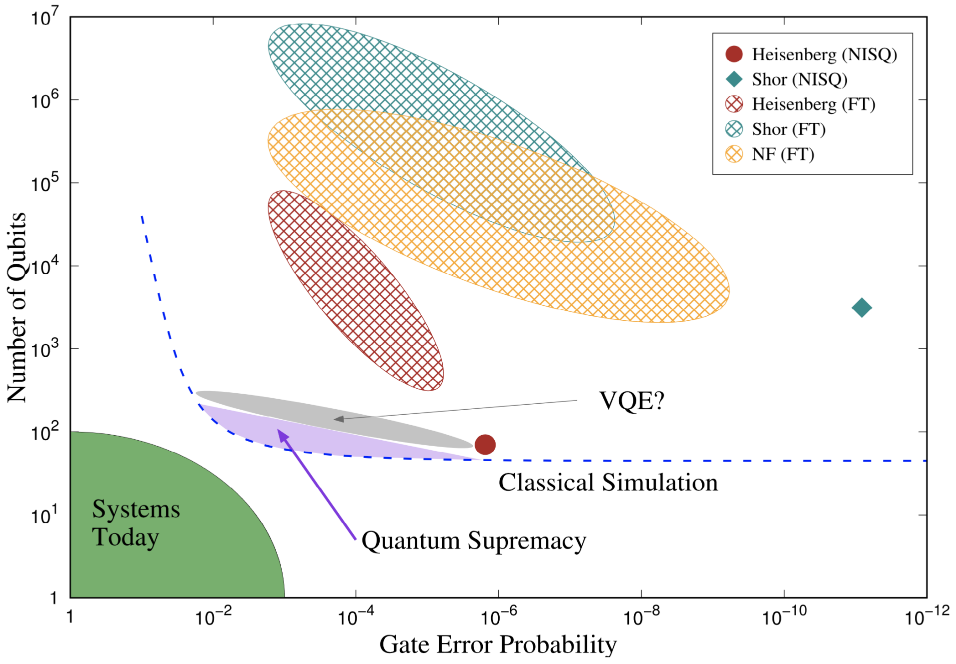


Fig. 2. *Performance space of quantum computers, measured by the error probability of each entangling gate in the horizontal axis (roughly inversely proportional to the total number of gates that can be executed on a NISQ machine), and the number of qubits in the system in the vertical axis. Blue dotted line approximately demarcates quantum systems that can be simulated using best classical computers, while the green colored region shows where the existing quantum computing systems with verified performance numbers lie (as of September 2018). Purple shaded region indicates computational tasks that accomplish the so-called "quantum supremacy," where the computation carried out by the quantum computer defies classical simulation regardless of its usefulness. The different shapes illustrate resource counts for solving various problems, with solid symbols corresponding to the exact entangling gate counts and number of qubits in NISQ machines, and shaded regions showing approximate gate error requirements and number of qubits for an FT implementation (not pictured are the regions where the error gets too close to the known fault-tolerance thresholds): cyan diamond and shaded region correspond to factoring a 1024-bit number using Shor's algorithm [14], magenta circle and shaded region represent simulation of a 72-spin Heisenberg model [20], and orange shaded region illustrates NF simulation [21].*

Figure: Credit: Maslov, Nam, and Kim. An Outlook for Quantum Computing. Proceedings of the IEEE. 2019.

Near-term intermediate-scale quantum (NISQ) computers

The limitations of near term quantum computers

- NISQ systems have limited number of qubits:
  No error correction.
  (In contrast, error corrected Shor's would need a million qubits.)
- NISQ systems have limited coherence time:
  Relative shallow depth of circuits.
  (In contrast, error corrected Shor's would need hundreds of millions of gates.)
- NISQ systems have limited operation accuracy

Table of contents

NISQ variational algorithms

Use a classical algorithm to train a "quantum neural network".

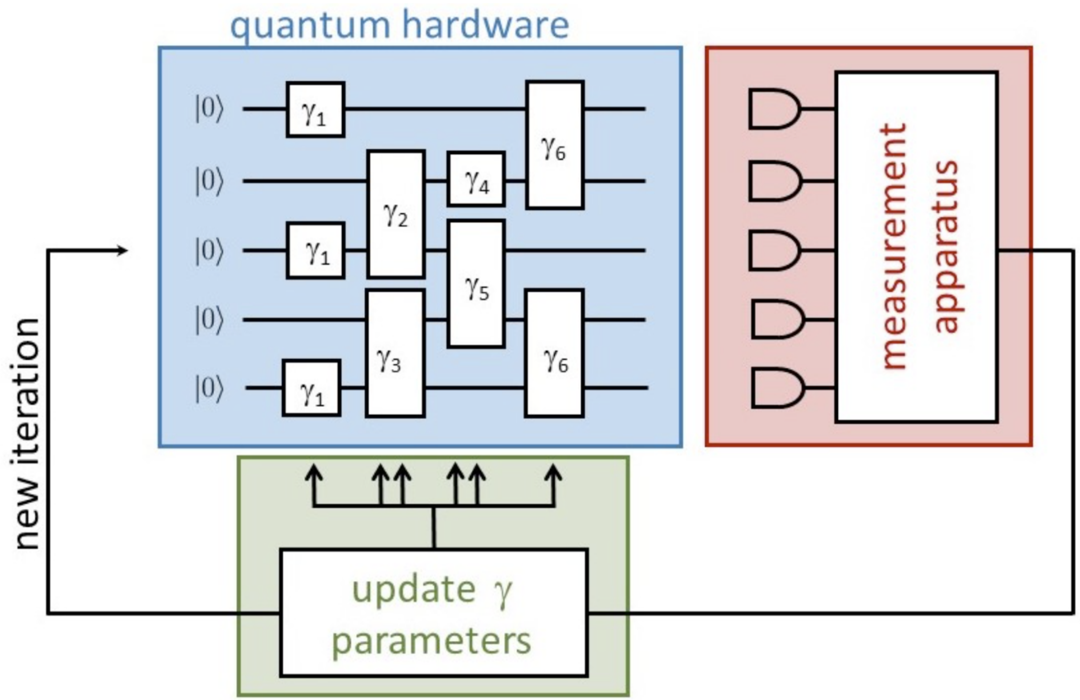


FIG. 1. Illustration of the three common steps of hybrid quantum-classical algorithms. These steps have to be repeated until convergence or when a sufficiently good quality of the solution is reached. 1) State preparation involving the quantum hardware capable of tunable gates characterized by parameters γ_n (blue), 2) measurement of the quantum state and evaluation of the objective function (red), 3) iteration of the

Figure:
Credit: [Guerreschi and Smel

NISQ variational algorithms

Use a classical algorithm to train a "quantum neural network".

1. Quantum computer prepares a quantum state that is a function of classical parameters.
2. Quantum computer measures quantum state to provide classical observations.
3. Classical computer uses observations to calculate an objective function.
4. Classical computer uses optimization routine to propose new classical parameters to maximize objective function.
5. Repeating steps 1 through 4, the algorithm leads to better approximations to underlying problem.

NISQ variational algorithms

Great! Can NISQ variational algorithms solve useful problems?

1. Variational quantum eigensolver (VQE):
   Simulate quantum mechanics
2. Quantum approximate optimization algorithm (QAOA):
   Approximate solutions to constraint satisfaction problems (CSPs) [Farhi et al., 2014]

Table of contents

Table of contents

Constraint satisfaction problem (CSP): MAX-CUT

- Given an arbitrary undirected graph $G = (V(G), E(G))$
- goal of MAX-CUT is to assign one of two partitions to each node so as to maximize the number of cuts

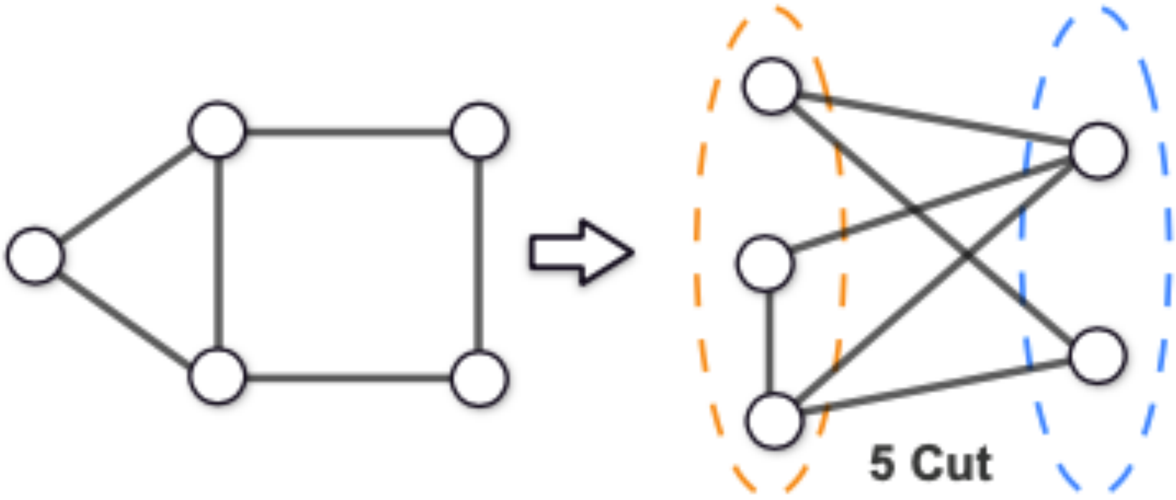


FIG. 39: An illustration of the MaxCut problem.

Figure: Credit: Quantum Algorithm Implementations for Beginners Coles.

Constraint satisfaction problem (CSP): MAX-CUT

- Given an arbitrary undirected graph
  $G = (V(G), E(G))$
- goal of MAX-CUT is to assign one of two partitions $\sigma_i \in \{-1, +1\}$ to each node $i \in V(G)$ so as to maximize the number of cuts
- Identical form to the MAX-SAT problem with objective function $C(\vec{\sigma})$:
  $$\max_{\vec{\sigma}} C(\vec{\sigma}) = \max_{\vec{\sigma}} \sum_{<jk>\in E(G)} C_{<jk>}(\vec{\sigma})$$
- But the constraints are now:
  $$C_{<jk>}(\vec{\sigma}) = \frac{1}{2}(1 - \sigma_j\sigma_k) = \begin{cases} 1 & \text{if } \sigma_j \text{ and } \sigma_k \text{ are different} \\ 0 & \text{if } \sigma_j \text{ and } \sigma_k \text{ are the same} \end{cases}$$

QAOA for MAX-CUT: general strategy

1. Each node in n nodes of the MAX-CUT graph corresponds to one of n qubits in the quantum circuit.
2. The state vector across the qubits $|\psi\rangle$ encodes a node partitioning $\vec{\sigma} \in \{-1, +1\}^n$
3. Put the initial state vector $|\psi_s\rangle$ in a superposition of all possible node partitionings
4. Need an operator (quantum gate) that encodes an edge $< jk > \in E(G)$
5. Provide classical parameters such that the classical computer can control quantum partitioning
6. Perform a series of operations parameterized by classical parameters $\vec{\gamma}$ and $\vec{\beta}$ such that the final state vector $|\psi(\vec{\gamma}, \vec{\beta})\rangle$ is a superposition of good partitionings
7. Optimize for a good set of $\vec{\gamma}$ and $\vec{\beta}$

QAOA for MAX-CUT: general strategy

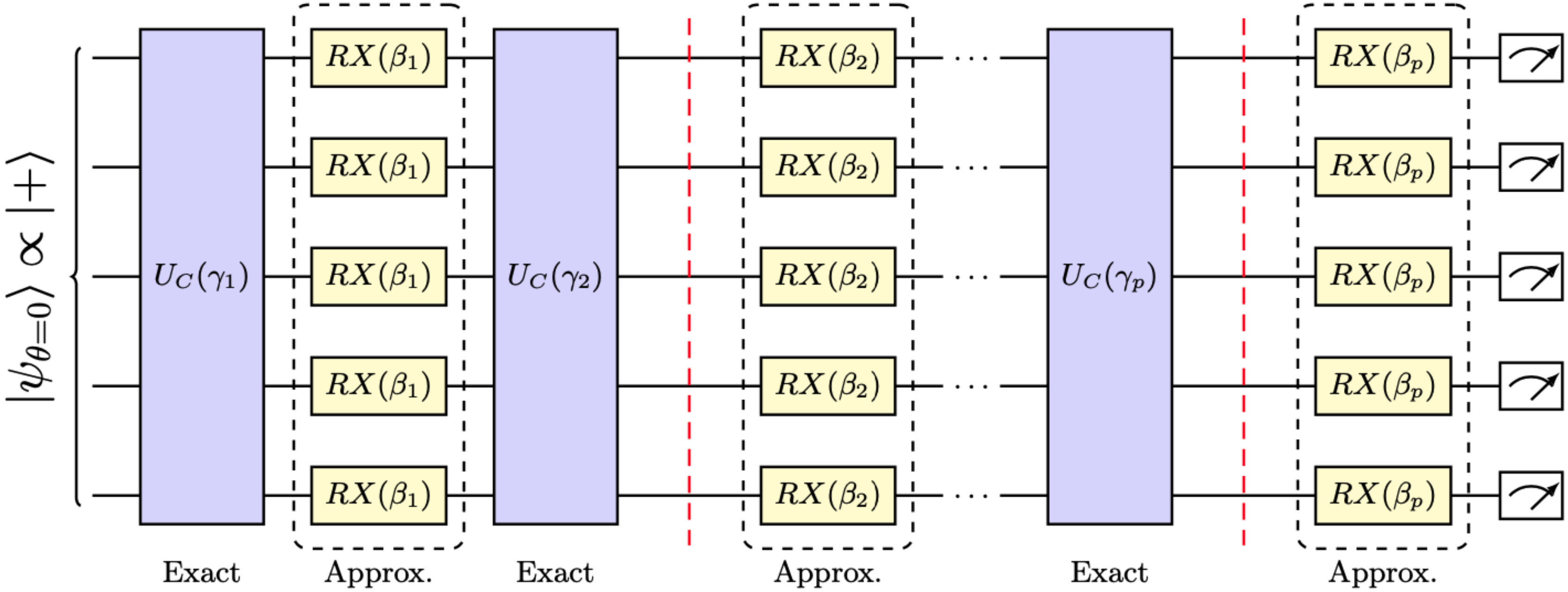


FIG. 1. A schematic representation of the QAOA circuit and our approach to simulating it. The input state is trivially initialized to $|+\rangle$. Next, at each p, the exchange of exactly (U_C, Sec. II B 1) and approximately ($RX(\beta) = e^{-i\beta X}$, Sec. II B 2) applicable gates is labeled. As noted in the main text, each (exact) application of the U_C gate leads to an increase in the number of hidden units by $|E|$ (the number of edges in the graph). In order to keep that number constant, we compress the number of hidden units (Sec. II C), indicated by red dashed lines after each U_C gate. The compression is repeated at each layer after the first, halving the number of hidden units each time.

Figure: Credit: Classical variational simulation of the Quantum Approximate Optimization Algorithm. Medvidovic and Carleo.

Table of contents

1. Each node in n nodes of the MAX-CUT graph corresponds to one of n qubits in the quantum circuit.

Let's use an $n = 3$ example in the figure.

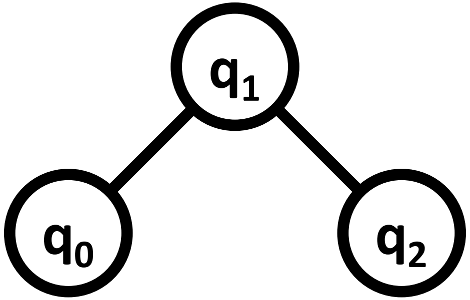


Figure: $G = (V(G), E(G)) = (\{q0, q1, q2\}, \{< q0q1 >, < q1q2 >\})$

$$|\psi\rangle = \alpha_0 |000\rangle + \alpha_1 |001\rangle ... + \alpha_7 |111\rangle = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ ... \\ \alpha_7 \end{bmatrix}$$

So now we have quantum amplitudes for each of the basis states.

Probability of measuring outcome z is $|\alpha_z|^2$. $\sum^{n-1} |\alpha_z|^2 = 1$

2. The state vector across the qubits $|\psi\rangle$ encodes a node partitioning

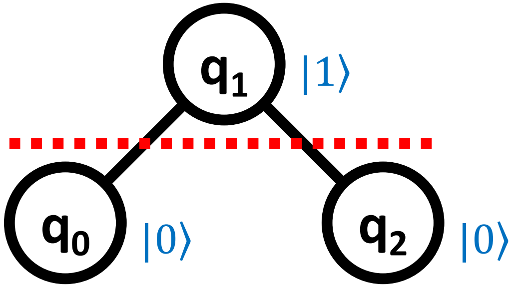


Figure: A max-cut corresponding to $|\psi\rangle = |010\rangle$.

- In our $n = 3$ running example, $|\psi\rangle = |010\rangle$, $\alpha_2 = 1$, $\alpha_0 = \alpha_1 = \alpha_3 = \alpha_4 = \alpha_5 = \alpha_6 = \alpha_7 = 0$, is a max-cut.
- The other max-cut is $|\psi\rangle = |101\rangle$.
- A superposition $|\psi\rangle = \frac{1}{\sqrt{2}}|010\rangle + \frac{1}{\sqrt{2}}|101\rangle$ would be an equal superposition of the two max cuts.

3. Put the initial state vector $|\psi_s\rangle$ in a superposition of all possible node partitionings

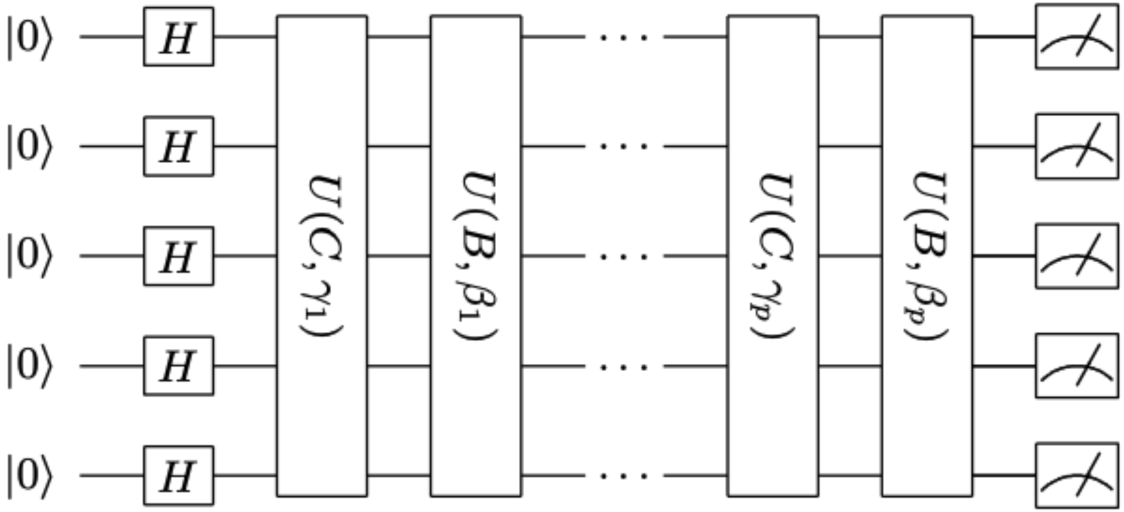


FIG. 2: Framework for a QAOA circuit. Each qubit begins with a Hadamard gate, and then $2p$ gates are performed alternating between applying Hamiltonian C and applying Hamiltonian B.

Figure: Credit: How many qubits are needed for quantum computational supremacy. Dalzell et al.

3. Put the initial state vector $|\psi_s\rangle$ in a superposition of all possible node partitionings

A superposition across all the bitstrings representing partitionings.

$$|\psi_s\rangle = |+\rangle^{\otimes n} = H^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{z=0}^{2^n-1} |z\rangle$$

3. Put the initial state vector $|\psi_s\rangle$ in a superposition of all possible node partitionings

In our $n = 3$ running example:

$$|+\rangle^{\otimes n} = H^{\otimes 3} |0\rangle^{\otimes 3} = \begin{bmatrix} \frac{+1}{\sqrt{2}} & \frac{+1}{\sqrt{2}} \\ \frac{+1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix}^{\otimes 3} \begin{bmatrix} 1 \\ 0 \end{bmatrix}^{\otimes 3} = \begin{bmatrix} \frac{+1}{\sqrt{2}} \begin{bmatrix} \frac{+1}{\sqrt{2}} & \frac{+1}{\sqrt{2}} \\ \frac{+1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} & \frac{+1}{\sqrt{2}} \begin{bmatrix} \frac{+1}{\sqrt{2}} & \frac{+1}{\sqrt{2}} \\ \frac{+1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \\ \frac{+1}{\sqrt{2}} \begin{bmatrix} \frac{+1}{\sqrt{2}} & \frac{+1}{\sqrt{2}} \\ \frac{+1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} & \frac{-1}{\sqrt{2}} \begin{bmatrix} \frac{+1}{\sqrt{2}} & \frac{+1}{\sqrt{2}} \\ \frac{+1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \end{bmatrix} \otimes$$

$$\begin{bmatrix} \frac{+1}{\sqrt{2}} & \frac{+1}{\sqrt{2}} \\ \frac{+1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}^{\otimes 3} = \frac{1}{\sqrt{8}} \sum_{z=0}^{7} |z\rangle = [\frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}, \frac{1}{\sqrt{8}}]^{\dagger}$$

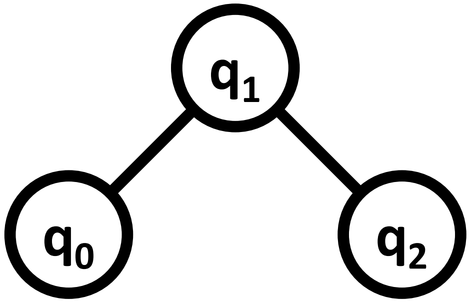
Table of contents

4. Need an operator (quantum gate) that encodes an edge $< jk > \in E(G)$

- In the Max-Cut type of CSP, constraints correspond to edges
- $C = \sum_{<jk> \in E(G)} C_{<jk>} = \sum_{<jk> \in E(G)} \frac{1}{2}(1 - \sigma_j^z \otimes \sigma_k^z)$
- σ_i^z is the Pauli-Z operator on qubit i
- $\sigma^z = \begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix}$
- Claim: ψ that maximizes $\langle \psi | C | \psi \rangle$ is the graph partition with the maximum cut.

4. Need an operator (quantum gate) that encodes an edge $< jk > \in E(G)$

In our $n = 3$ running example:



$$G = (V(G), E(G))$$
$$= (\{q0, q1, q2\}, \{< q0q1 >, < q1q2 >\}$$

$$C = \sum_{<jk>\in E(G)} \frac{1}{2}(1 - \sigma_j^z \otimes \sigma_k^z)$$
$$= \frac{1}{2}(1 - \sigma_0^z \otimes \sigma_1^z) + \frac{1}{2}(1 - \sigma_1^z \otimes \sigma_2^z)$$
$$= \frac{1}{2}(I - \sigma^z \otimes \sigma^z \otimes I) + \frac{1}{2}(I - I \otimes \sigma^z \otimes \sigma^z)$$
$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

4. Need an operator (quantum gate) that encodes an edge $< jk > \in E(G)$

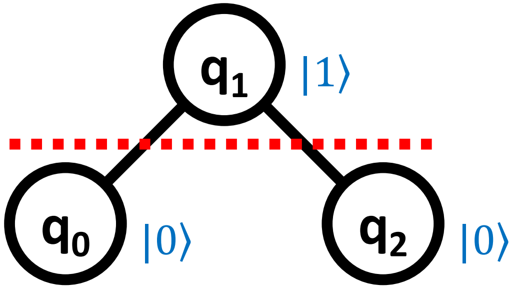


Figure: A max-cut corresponding to $|\psi\rangle = |010\rangle$.

Claim: ψ that maximizes $\langle\psi| C |\psi\rangle$ is the graph partition with a maximum cut.

$$\langle 010| C |010\rangle = \begin{bmatrix} 0\\0\\1\\0\\0\\0\\0\\0 \end{bmatrix}^{\dagger} \begin{bmatrix} 0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&2&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&2&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0 \end{bmatrix} \begin{bmatrix} 0\\0\\1\\0\\0\\0\\0\\0 \end{bmatrix} = 2$$

4. Need an operator (quantum gate) that encodes an edge $< jk > \in E(G)$

- $\langle 010| C |010\rangle = \begin{bmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix}^{\dagger} \begin{bmatrix}0&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&2&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&2&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0\end{bmatrix} \begin{bmatrix}0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix} = 2$
- Note the size of state vector $|\psi\rangle$ is 2^n.
- Size of constraint matrix C is $2^n \times 2^n$.
- We wouldn't want to construct C explicitly, but it can be created efficiently using gates and tensor products.
- We will enlist a quantum computer to create $|\psi\rangle$ and C.

Table of contents

5. Provide classical parameters such that the classical computer can control quantum partitioning

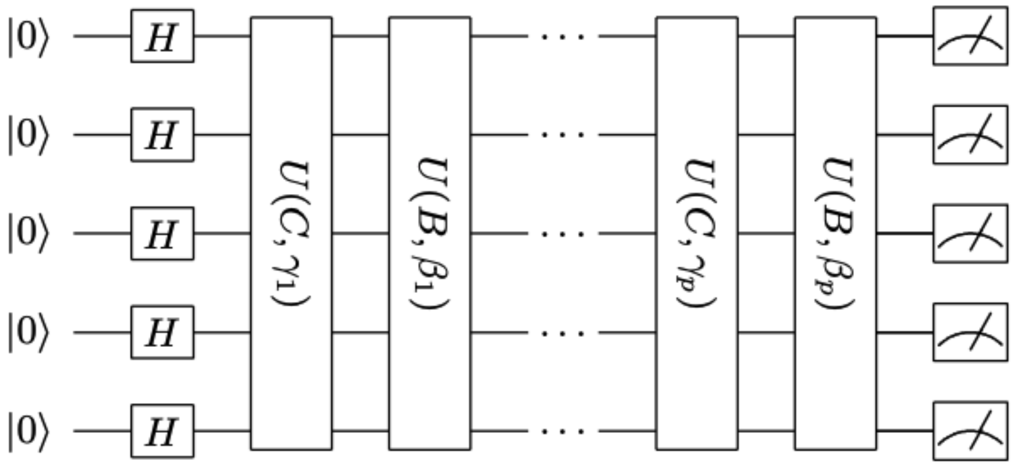


FIG. 2: Framework for a QAOA circuit. Each qubit begins with a Hadamard gate, and then $2p$ gates are performed alternating between applying Hamiltonian C and applying Hamiltonian B.

Figure: Credit: How many qubits are needed for quantum computational supremacy. Dalzell et al.

5. Provide classical parameters such that the classical computer can control quantum partitioning

- A parameter p that controls how many iterations of algorithm and how complete of a graph to see.
- Do optimization across $2p$-dimensional vector of $\vec{\gamma}$ and $\vec{\beta}$ parameters
- $(\vec{\gamma}, \vec{\beta}) = (\gamma_1, \beta_1, ... \gamma_p, \beta_p)$
- $\gamma_i \in [0, 2\pi]$
- $\beta_i \in [0, \pi]$

6. Perform a series of operations parameterized by classical parameters $\vec{\gamma}$ and $\vec{\beta}$ such that the final state vector $|\psi(\vec{\gamma}, \vec{\beta})\rangle$ is a superposition of good partitionings

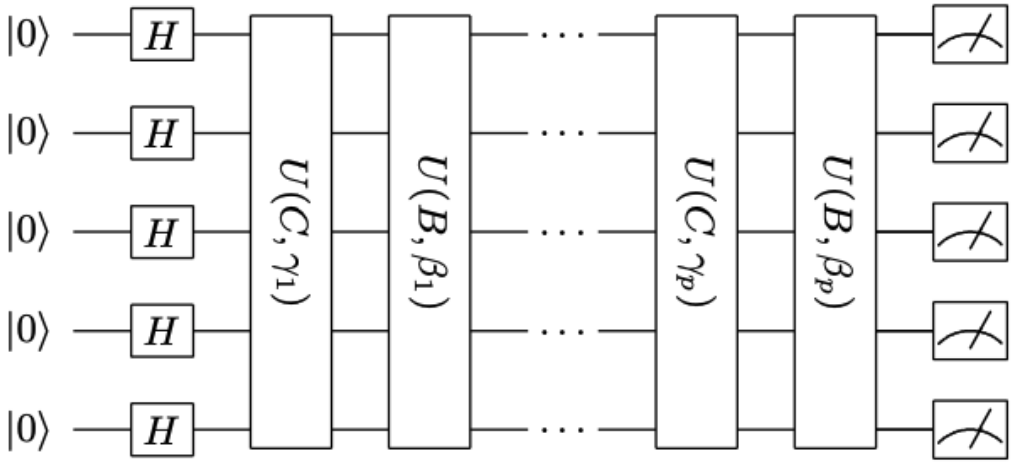


FIG. 2: Framework for a QAOA circuit. Each qubit begins with a Hadamard gate, and then $2p$ gates are performed alternating between applying Hamiltonian C and applying Hamiltonian B.

Figure: Credit: How many qubits are needed for quantum computational supremacy. Dalzell et al.

6. Perform a series of operations parameterized by classical parameters $\vec{\gamma}$ and $\vec{\beta}$ such that the final state vector $|\psi(\vec{\gamma}, \vec{\beta})\rangle$ is a superposition of good partitionings

$U(C, \gamma), U(B, \beta)$ are $2^n \times 2^n$ linear operators (Unitary matrices)

1. Problem Hamiltonian enforces constraints.
   A product across all the graph edges.
   $U(C, \gamma) = e^{-i\gamma C} = \prod_{<jk>\in E(G)} e^{-i\gamma C_{<jk>}}$

2. Admixing Hamiltonian perturbs the assignments.
   A product across all the qubits representing graph vertices.
   $U(B, \beta) = e^{-i\beta B} = \prod_{q\in V(G)} e^{-i\beta \sigma_q^x}$

6. Perform a series of operations parameterized by classical parameters $\vec{\gamma}$ and $\vec{\beta}$ such that the final state vector $|\psi(\vec{\gamma}, \vec{\beta})\rangle$ is a superposition of good partitionings

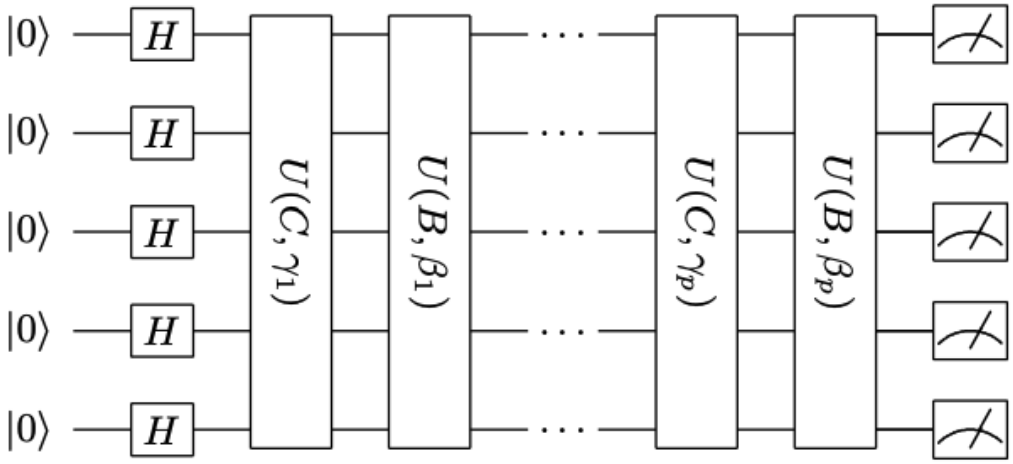


FIG. 2: Framework for a QAOA circuit. Each qubit begins with a Hadamard gate, and then $2p$ gates are performed alternating between applying Hamiltonian C and applying Hamiltonian B.

Figure: Credit: How many qubits are needed for quantum computational supremacy. Dalzell et al.

6. Perform a series of operations parameterized by classical parameters $\vec{\gamma}$ and $\vec{\beta}$ such that the final state vector $|\psi(\vec{\gamma}, \vec{\beta})\rangle$ is a superposition of good partitionings

- Create ansatz states $|\psi(\vec{\gamma}, \vec{\beta})\rangle$
- $|\psi(\vec{\gamma}, \vec{\beta})\rangle = U(B, \beta_p)U(C, \gamma_p)...U(B, \beta_1)U(C, \gamma_1)\,|\psi_s\rangle = \prod_{i=1}^{p}\left(\prod_{q \in V(G)} e^{-i\beta_i \sigma_q^x} \prod_{<jk> \in E(G)} e^{-i\gamma_i C_{<jk>}}\right)|+\rangle^{\otimes n}$

7. Optimize for a good set of $\vec{\gamma}$ and $\vec{\beta}$

- Measure this state to compute the objective function. That is, given the current set of parameters $\vec{\gamma}$ and $\vec{\beta}$, how much of the CSP is satisified
- Use a classical optimization algorithm such as Nelder-Mead to maximize $F_p(\vec{\gamma}, \vec{\beta}) = \langle \psi(\vec{\gamma}, \vec{\beta}) | \, C \, | \psi(\vec{\gamma}, \vec{\beta}) \rangle$

Table of contents

Evaluation of QAOA for NISQ: Number of iterations?

- Let M_p be the maximum of F_p over the angles: $M_p = \max_{\vec{\gamma},\vec{\beta}} F_p(\vec{\gamma}, \vec{\beta})$
- QAOA needs a big parameter p to see the whole graph
- As $p \to \infty$, $\lim_{p\to\infty} M_p = \max_z C(z)$

Evaluation of QAOA for NISQ: Number of iterations?

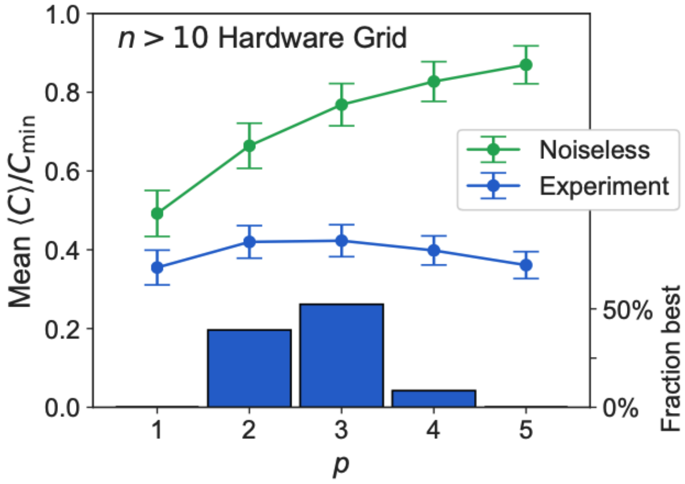


FIG. 5. QAOA performance as a function of depth, p. In ideal simulation, increasing p increases the quality of the solution. For experimental Hardware Grid results, we observe increased performance for $p > 1$ both as measured by the mean over all 10 instances studied for each value of $n \in [11, 23]$ (lines) and statistics of which p value maximizes performance on a per-instance basis (histogram). At larger p, errors overwhelm the theoretical performance increase.

Figure: Credit: [Harrigan et al., 2021]

Evaluation of QAOA for NISQ: Number of qubits?

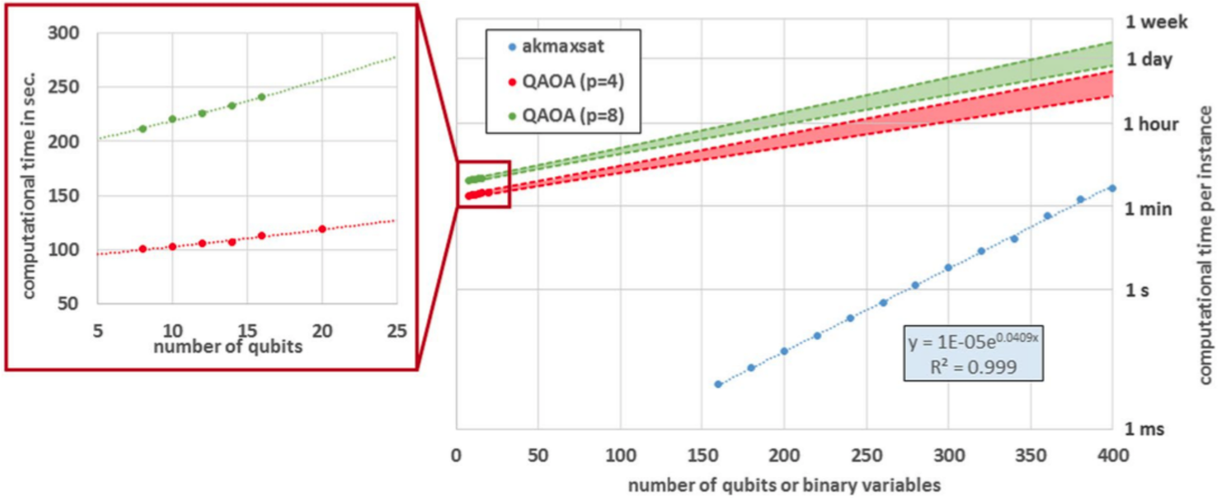


Figure 2. Main panel: Computational cost of solving a single Max-Cut instance on random 3-regular graphs. Blue markers correspond to the classical baseline (AKMAXSAT solver) while red and green marks correspond to the experimental time required by the quantum algorithm QAOA, with $p = 4$ and $p = 8$ respectively. The error bars for the single data points are smaller than the markers (see Supplementary Information). Notice that in the time needed by QAOA to partition graphs with 20 vertices, AKMAXSAT partitions graphs about 20 times larger. The blue dashed line is the result of a fitting procedure with an exponential function. The red and green areas are associated with a 95% confidence interval for the prediction of the QAOA cost based on a linear regression of $\log_{10}(T)$ as a function of the number of qubits (here T is the computational time per instance). This extrapolation should be seen as suggesting a qualitative behavior due to the uncertainty in the extrapolation from relatively small system sizes. Insert panel: Magnification of QAOA datapoints. Notice that exponential curves, and smooth curves in general, locally resemble straight lines and this makes it difficult to exclude other functional forms for the extrapolation. It is, however, believed that even quantum computers will not be able to solve NP-hard problems in polynomial time.

Figure: Credit: [Guerreschi and Matsuura, 2019]

Evaluation of QAOA for NISQ: Number of constraints?

- Findings for MAX-CUT on connected 3-regular graphs [Farhi et al., 2014]
  1. For p = 1, QAOA will always produce a cut whose size is at least 0.6924 times the size of the optimal cut.
  2. This was the best known possible approximation for a few months until classical algorithm found.
  3. For p = 2, the approximation ratio becomes 0.7559 and grows depending on the type of graph.
- Difficulty of solving problems with higher constraint ratios [Akshay et al., 2020]

Evaluation of QAOA for NISQ: Optimization method?

- Optimization using gradients [Guerreschi and Smelyanskiy, 2017].
- Optimization using reinforcement learning [Khairy et al., 2020]

Evaluation of QAOA for NISQ: Generalizations?

- Can be generalized to tackle weighted graphs [Willsch et al., 2020].
- Can be generalized to solve related problems [Hadfield et al., 2019].
- Factoring [Anschuetz et al., 2019].

Read also

- Primary sources: [Farhi et al., 2014]
- Additional source: [Wang and Abdullah, 2018]

Akshay, V., Philathong, H., Morales, M. E. S., and Biamonte, J. D. (2020).
Reachability deficits in quantum approximate optimization.
Phys. Rev. Lett., 124:090504.

Anschuetz, E., Olson, J., Aspuru-Guzik, A., and Cao, Y. (2019).
Variational quantum factoring.
In Feld, S. and Linnhoff-Popien, C., editors, *Quantum Technology and Optimization Problems*, pages 74–85, Cham. Springer International Publishing.

Córcoles, A. D., Kandala, A., Javadi-Abhari, A., McClure, D. T., Cross, A. W., Temme, K., Nation, P. D., Steffen, M., and Gambetta, J. M. (2020).
Challenges and opportunities of near-term quantum computing systems.
Proceedings of the IEEE, 108(8):1338–1352.

Farhi, E., Goldstone, J., and Gutmann, S. (2014).
A quantum approximate optimization algorithm.
arXiv preprint arXiv:1411.4028.

Guerreschi, G. G. and Matsuura, A. Y. (2019).
QAOA for max-cut requires hundreds of qubits for quantum speed-up.
Scientific Reports, 9(1):6903.

Guerreschi, G. G. and Smelyanskiy, M. (2017).
Practical optimization for hybrid quantum-classical algorithms.
arXiv preprint arXiv:1701.01450.

Hadfield, S., Wang, Z., O'Gorman, B., Rieffel, E. G., Venturelli, D., and Biswas, R. (2019).
From the quantum approximate optimization algorithm to a quantum alternating operator ansatz.

Algorithms, 12(2):34.

Harrigan, M. P., Sung, K. J., Neeley, M., Satzinger, K. J., Arute, F., Arya, K., Atalaya, J., Bardin, J. C., Barends, R., Boixo, S., Broughton, M., Buckley, B. B., Buell, D. A., Burkett, B., Bushnell, N., Chen, Y., Chen, Z., Chiaro, B., Collins, R., Courtney, W., Demura, S., Dunsworth, A., Eppens, D., Fowler, A., Foxen, B., Gidney, C., Giustina, M., Graff, R., Habegger, S., Ho, A., Hong, S., Huang, T., Ioffe, L. B., Isakov, S. V., Jeffrey, E., Jiang, Z., Jones, C., Kafri, D., Kechedzhi, K., Kelly, J., Kim, S., Klimov, P. V., Korotkov, A. N., Kostritsa, F., Landhuis, D., Laptev, P., Lindmark, M., Leib, M., Martin, O., Martinis, J. M., McClean, J. R., McEwen, M., Megrant, A., Mi, X., Mohseni, M., Mruczkiewicz, W., Mutus, J., Naaman, O., Neill, C., Neukart, F., Niu, M. Y., O'Brien, T. E., O'Gorman, B., Ostby, E., Petukhov, A., Putterman, H., Quintana, C., Roushan, P., Rubin, N. C., Sank, D., Skolik, A., Smelyanskiy, V., Strain, D., Streif, M., Szalay, M., Vainsencher, A., White, T., Yao, Z. J., Yeh, P., Zalcman, A., Zhou, L., Neven, H., Bacon, D., Lucero, E., Farhi, E., and Babbush, R. (2021).
Quantum approximate optimization of non-planar graph problems on a planar superconducting processor.
Nature Physics, 17(3):332–336.

Khairy, S., Shaydulin, R., Cincio, L., Alexeev, Y., and Balaprakash, P. (2020).
Learning to optimize variational quantum circuits to solve combinatorial problems.
In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2367–2375. AAAI Press.

Nielsen, M. A. and Chuang, I. L. (2011).
Quantum Computation and Quantum Information: 10th Anniversary Edition.
Cambridge University Press, USA, 10th edition.

Resch, S. and Karpuzcu, U. R. (2019).

Quantum computing: An overview across the system stack.

Wang, Q. and Abdullah, T. (2018).
An introduction to quantum optimization approximation algorithm.

Willsch, M., Willsch, D., Jin, F., De Raedt, H., and Michielsen, K. (2020).
Benchmarking the quantum approximate optimization algorithm.
Quantum Information Processing, 19(7):197.