CPEN 400Q Lecture 15
Solving combinatorial optimization problems with the Quantum Approximate Optimization Algorithm (part 1)

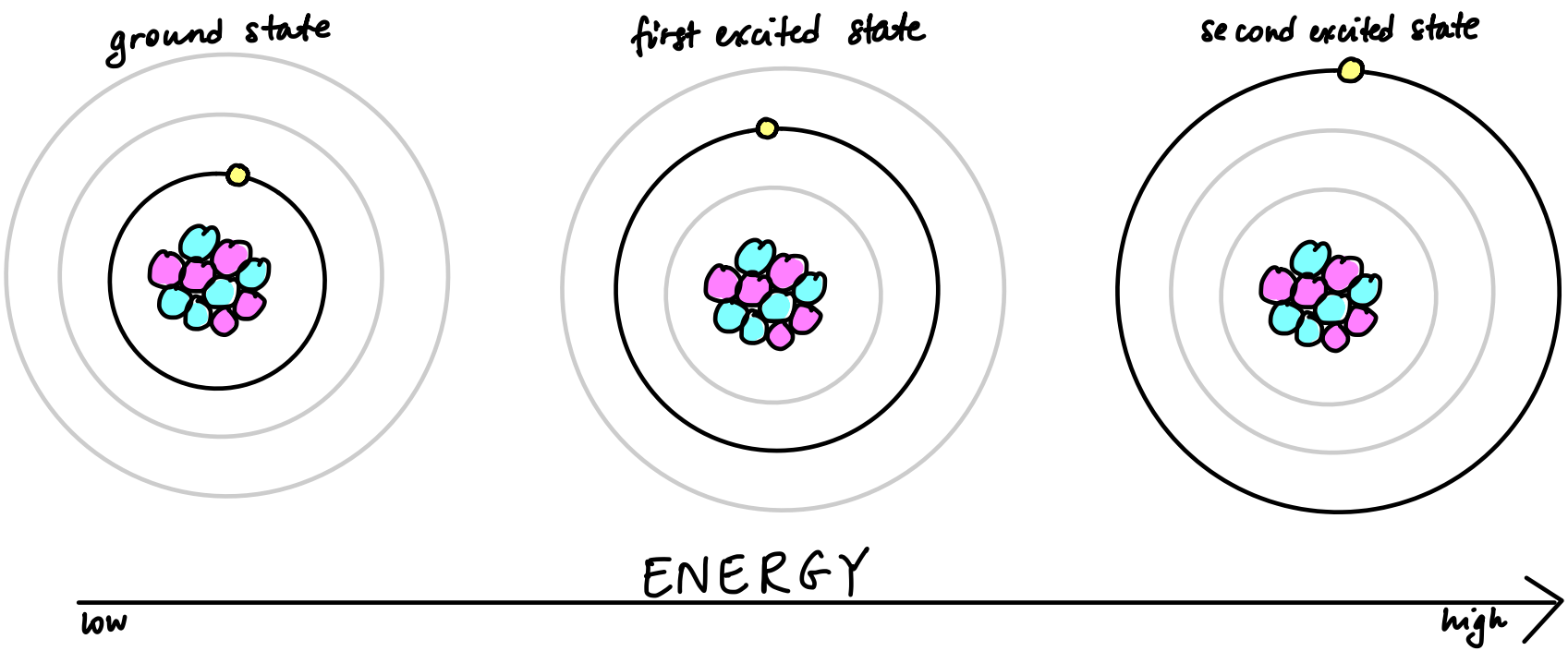
Monday 6 March 2023

Announcements

- Quiz 6 end of class today
- Assignment 2 due 13 March at 23:59
- Updated class schedule:
  - Friday March 10: Zoom (same link)
  - Monday March 13: in person (quiz 7)
  - Friday March 17: pre-recorded "infotainment" lecture about compilation
- Meetings *this week* for project prototypes on Zoom

Last time

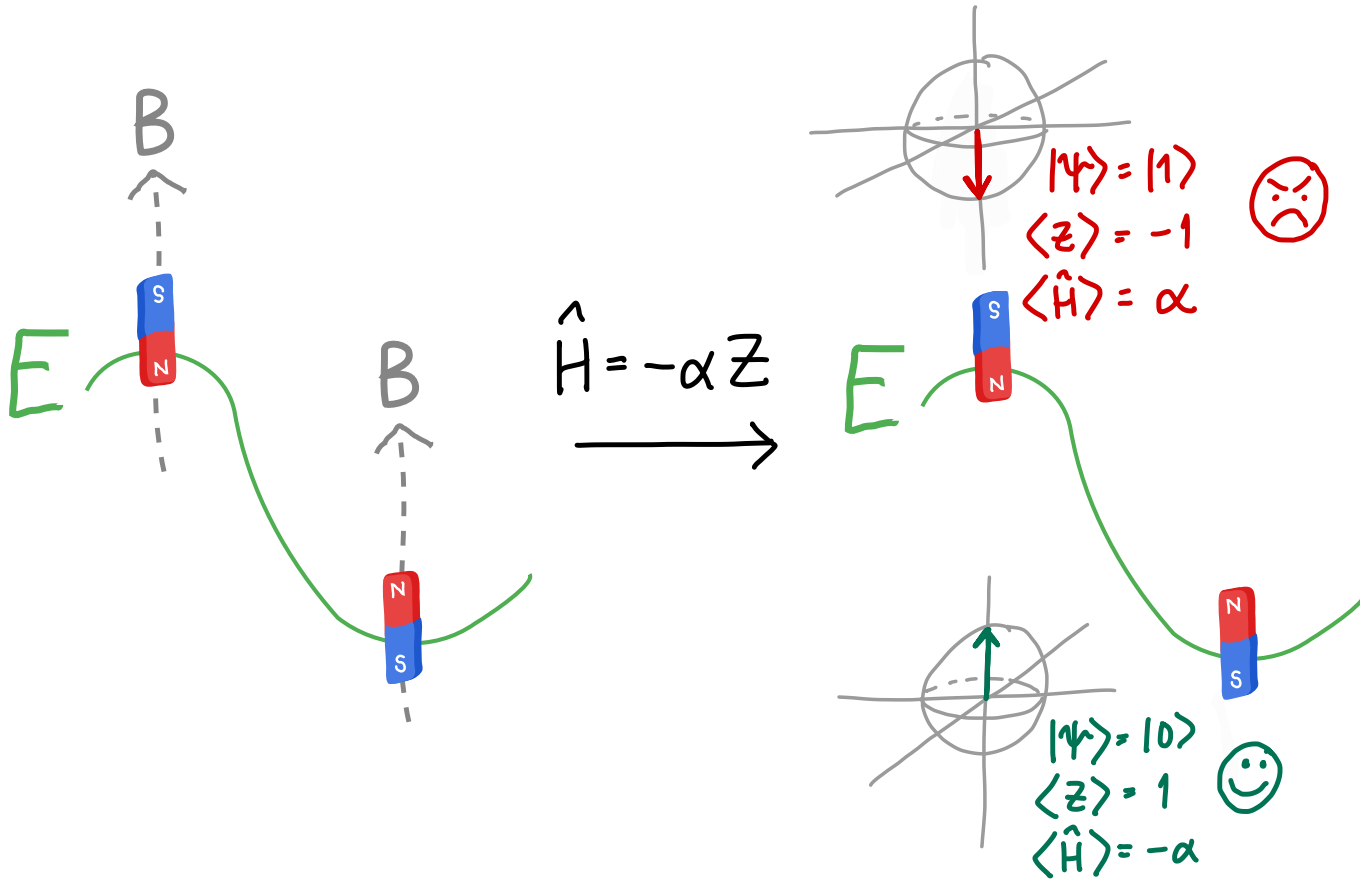
We tackled the problem of computing the *ground state* and *ground state energy* of a physical system.



Last time

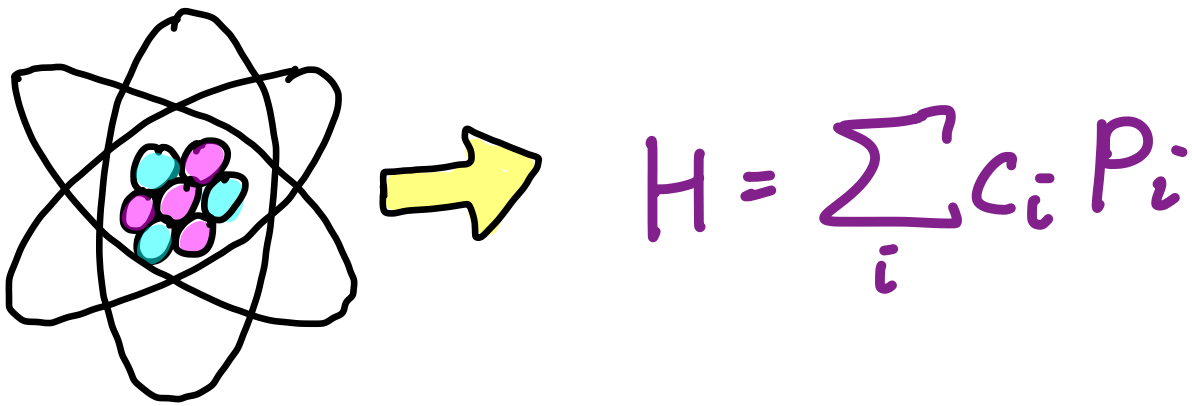
A **Hamiltonian** is an operator that describes the energy of a quantum system.

To run on a quantum computer, we *map* the system to an equivalent qubit-based Hamiltonian.



Last time

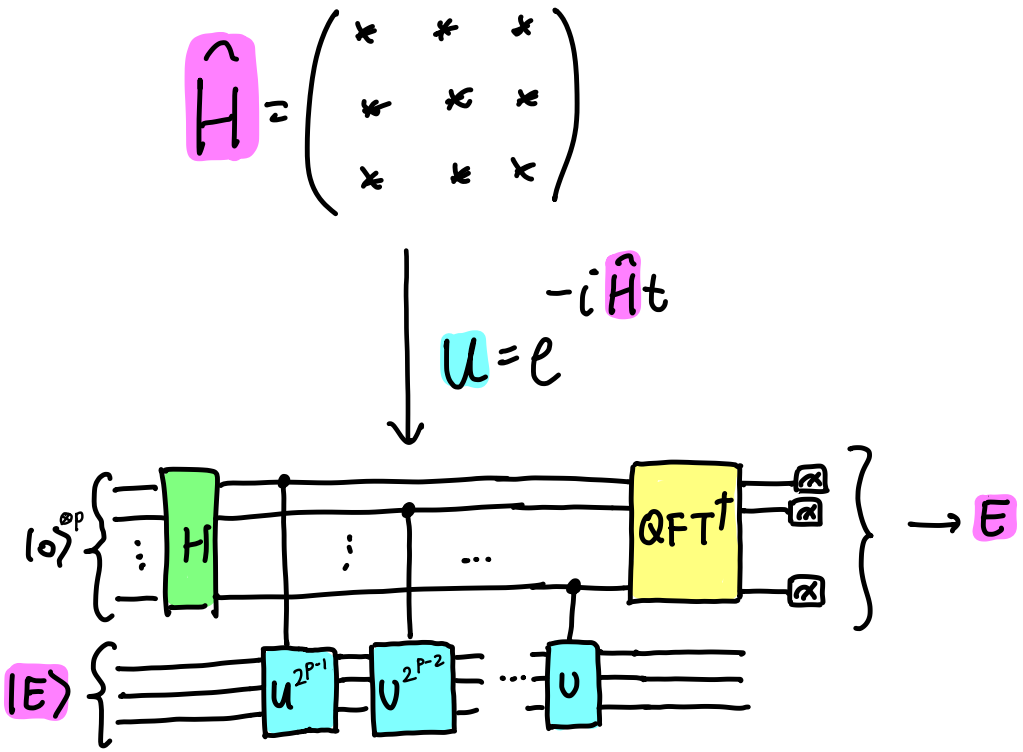
Any qubit Hamiltonian can be expressed as a linear combination of multi-qubit Pauli observables.



The Hamiltonian is a matrix. Its **spectrum** (**eigenvalues** and **eigenstates**) represents the possible measurement outcomes, just like single Paulis.

Last time

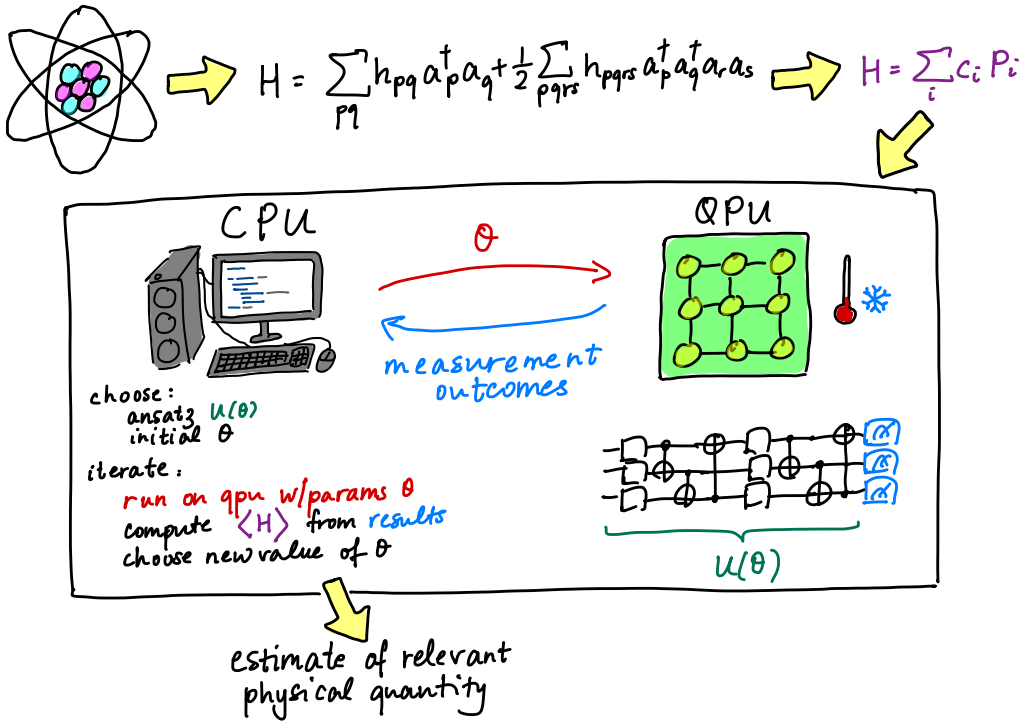
We can run *quantum phase estimation* to get the energy spectrum.



But this is not practical on near-term quantum devices.

Last time

The *variational quantum eigensolver* (VQE) is more practical.



The optimization works because of the *variational principle*:

for *any other state* $|\psi\rangle$.

Last time

You were given the Hamiltonian for a *deuteron* represented by a 4-qubit system.

I told you its ground state had the form

$$|\psi_g\rangle = a|1000\rangle + b|0100\rangle + c|0010\rangle + d|0001\rangle$$

Challenge:

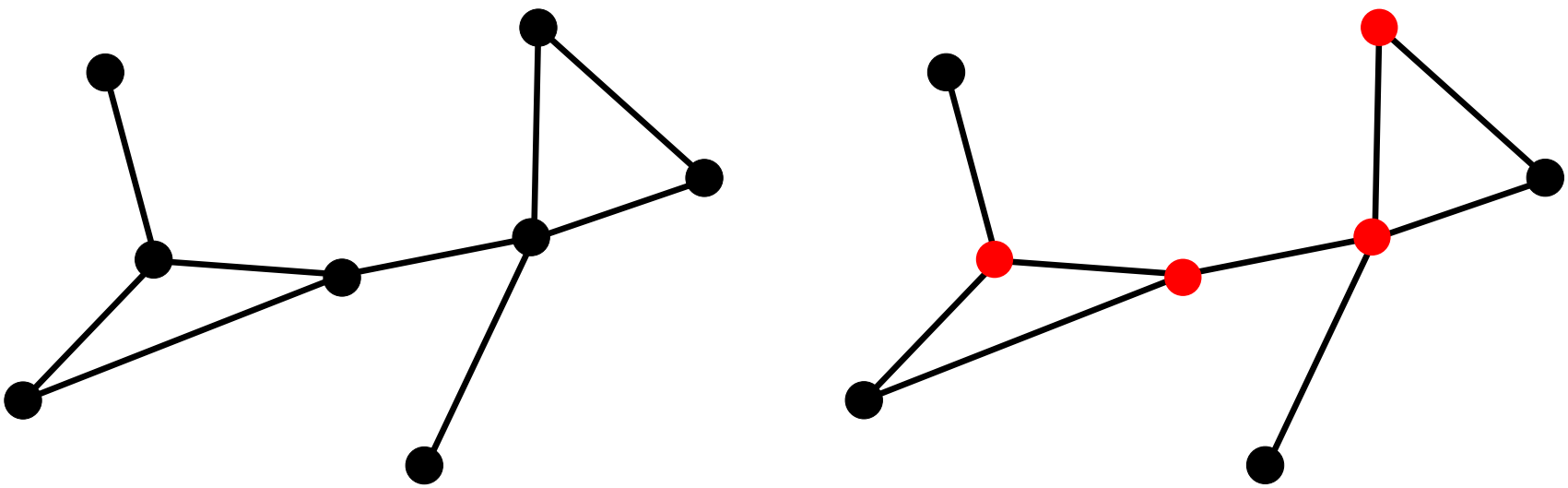
- Design a circuit ansatz that produces states that look like this
- Run VQE to determine the ground state energy

Learning outcomes

- Describe the underlying ideas of adiabatic quantum computation and QAOA
- Convert an optimization problem over binary variables into an equivalent problem over qubits.

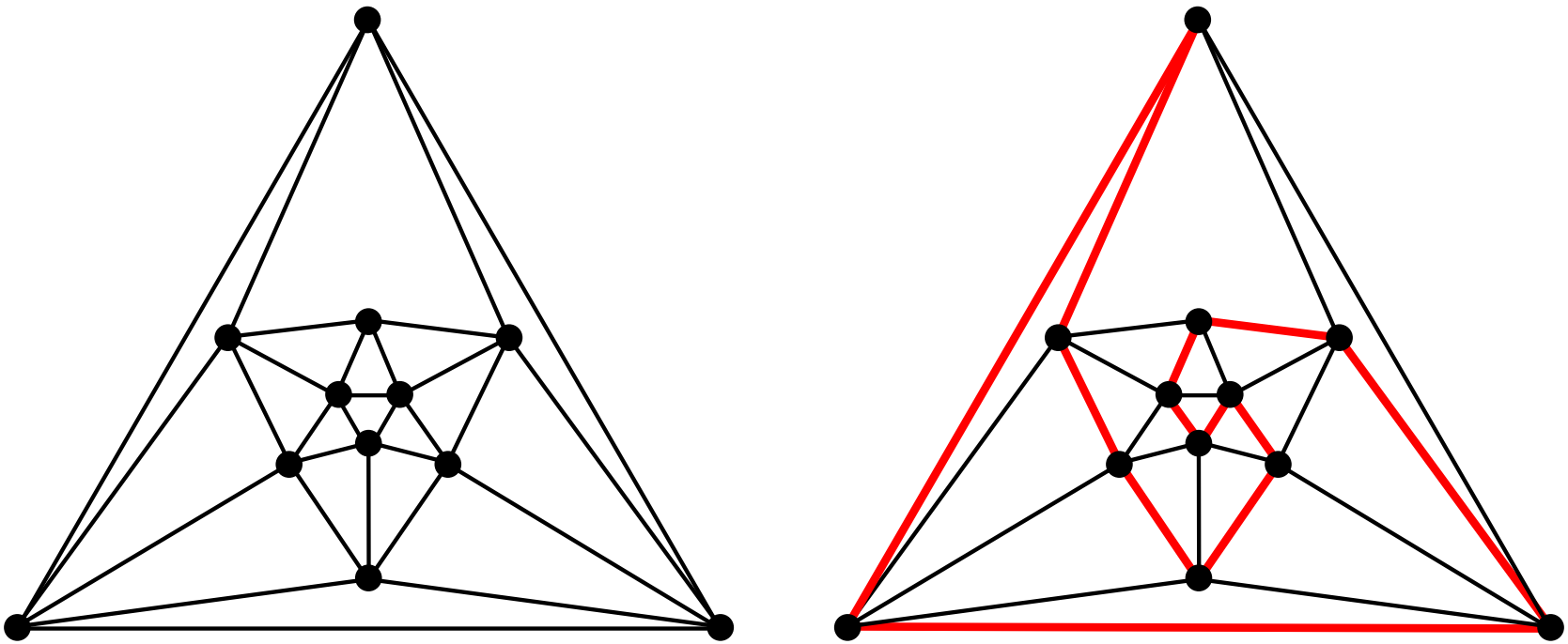
Combinatorial optimization

Example: Given a graph $G = (V, E)$, what is the *smallest number of vertices* you can colour such that every edge in the graph is attached to at least one coloured vertex?



Combinatorial optimization

Example: Given a graph, can we find a path through it that visits every node *exactly once* and returns to the starting point?



(In graph theory terms, can we find a *Hamiltonian cycle*?)

Combinatorial optimization

Example: You have the opportunity to purchase 100 units of stocks from a fixed list of assets. You know the average returns of each stock, and their covariances.

Stock	Avg. return
AAA	3.44 %
BBB	2.21 %
CCC	-0.28 %
⋮	⋮

Cov.	AAA	BBB	⋯
AAA	0.0038	0.002	⋯
BBB	0.002	-0.006	⋯
CCC	0.014	-0.0008	⋯
⋮	⋮	⋮	⋱

Suppose you're restricted to buying no more than 5 of any stock.

Which stocks, and how many of each, should you purchase, to maximize your profits?

Adiabatic quantum computing (AQC)

The structure of a classical optimization problem is something like:

$$\min_{\vec{x}} \ \text{cost}(\vec{x}) \quad \text{subject to constraints}(\vec{x})$$

where $\vec{x}$ is a multi-dimensional vector of variables of the problem.

Adiabatic quantum computing (AQC)

The structure of a classical optimization problem is something like:

$$\min_{\vec{x}} \ \text{cost}(\vec{x}) \quad \text{subject to constraints}(\vec{x})$$

where $\vec{x}$ is a multi-dimensional vector of variables of the problem.

In a physical context, optimization can be interpreted as an **energy minimization problem**.

Optimization	**Physical system**
$\vec{x}$	State of the system
$\text{cost}(\vec{x})$	Hamiltonian
Optimum $\vec{x^*}$	Ground state
$\text{cost}(\vec{x^*})$	Ground state energy

Adiabatic quantum computing (AQC)

Recall that every unitary U is associated with a Hamiltonian $\hat{H}$ under the correspondence

We know that we can use gate model QC to *simulate* the evolution of a Hamiltonian.

Instead of simulating the Hamiltonians, **adiabatic quantum computing** works with them directly to perform computations. It is generally used to solve **optimization problems**.

The adiabatic theorem

Theorem:

"A physical system remains in its instantaneous eigenstate if a given perturbation is acting on it slowly enough and if there is a gap between the eigenvalue and the rest of the Hamiltonian's spectrum."

What we can take from this:

If we initialize a system in the lowest energy state and perturb it slowly enough, it will remain in the lowest energy state (with respect to the changed system)

Adiabatic quantum computing (AQC)

Let H_c be a **cost Hamiltonian** whose ground state represents the solution to a problem of interest.

Let H_m be a **mixer Hamiltonian** whose ground state can be easily prepared.

Adiabatic evolution is expressed mathematically as the function

The parameter s is representative of time; s goes from 0 to 1; $A(s)$ decreases to 0 with time and $B(s)$ increases from 0.

Quantum annealing

D-Wave makes quantum annealers: these are a physical implementation of AQC for a limited set of Hamiltonians.

Image credit:
www.dwavesys.com/tutorials/background-reading-series/introduction-d-wave-quantum-hardware

Quantum approximate optimization algorithm (QAOA)

QAOA is a gate-model algorithm that can obtain approximate solutions to combinatorial optimization problems.

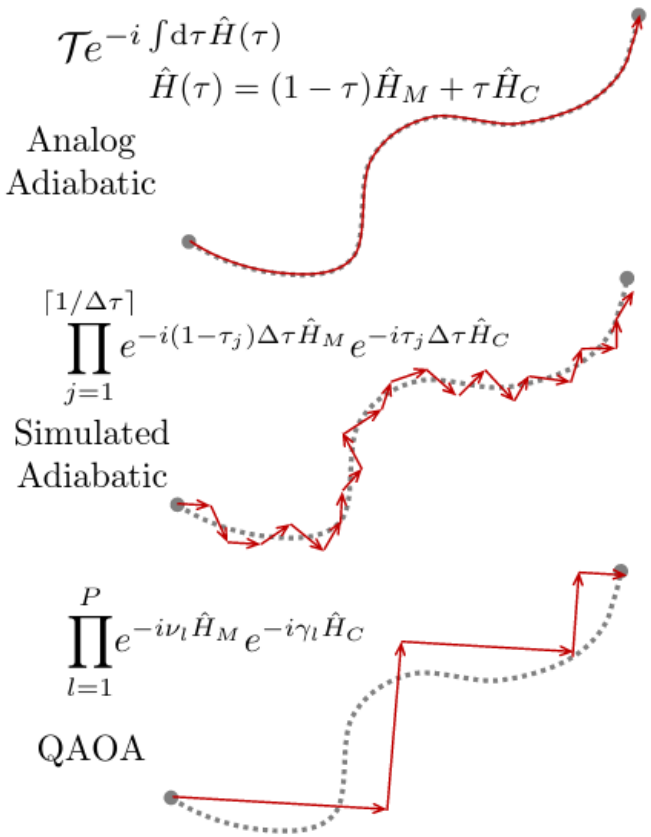
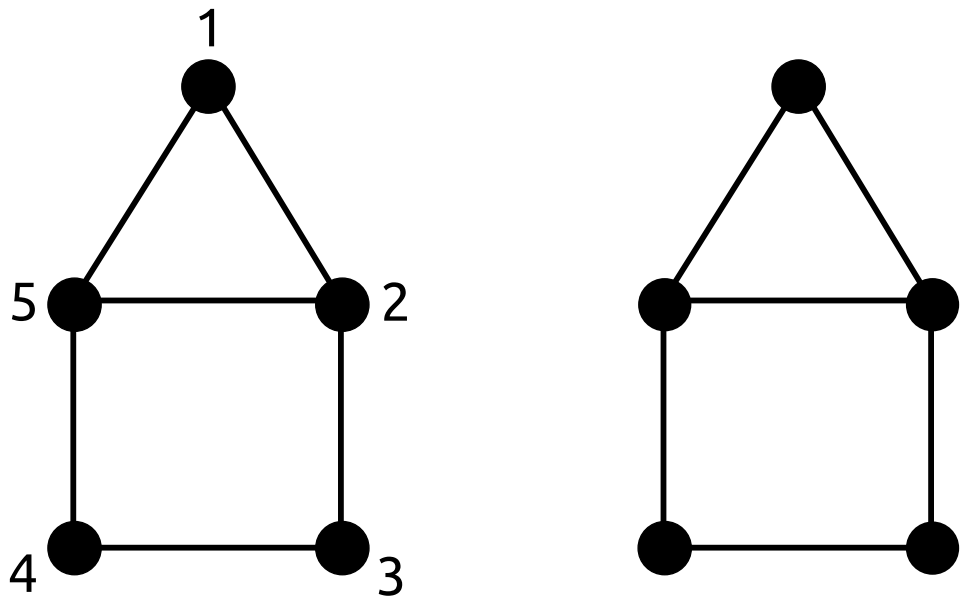


Image credit: G. Verdon, M. Broughton, J. Biamonte. *A quantum algorithm to train neural networks using low-depth circuits.* `https://arxiv.org/abs/1712.05304`

Motivating example: minimum vertex cover

Vertex cover: given a graph $G = (V, E)$ find the *minimum number* of coloured vertices such that every edge is adjacent to a coloured vertex. This problem is **NP-hard**.

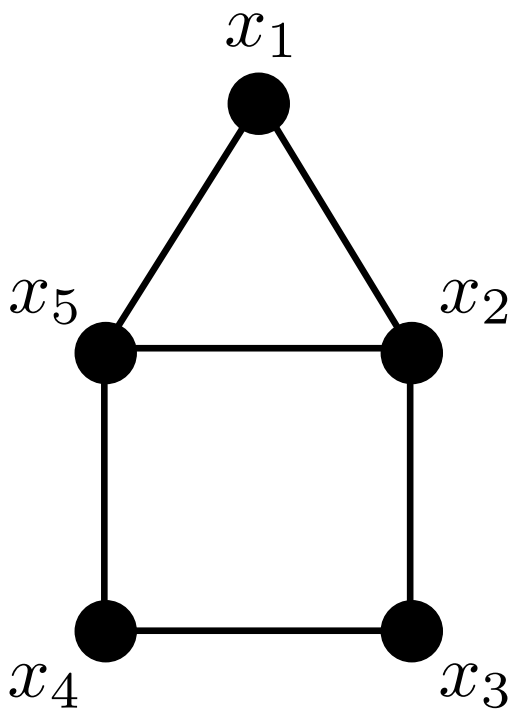


We will:

- define a cost function that is minimized for a valid colouring
- map it to a Hamiltonian
- run a quantum algorithm to find the ground state

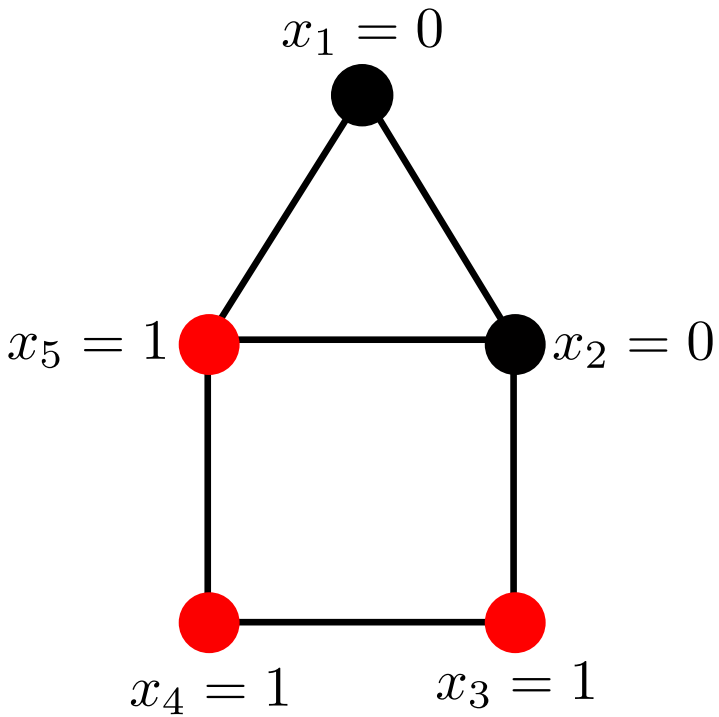
Motivating example: minimum vertex cover

Whether or not a vertex is coloured is a *binary variable*.



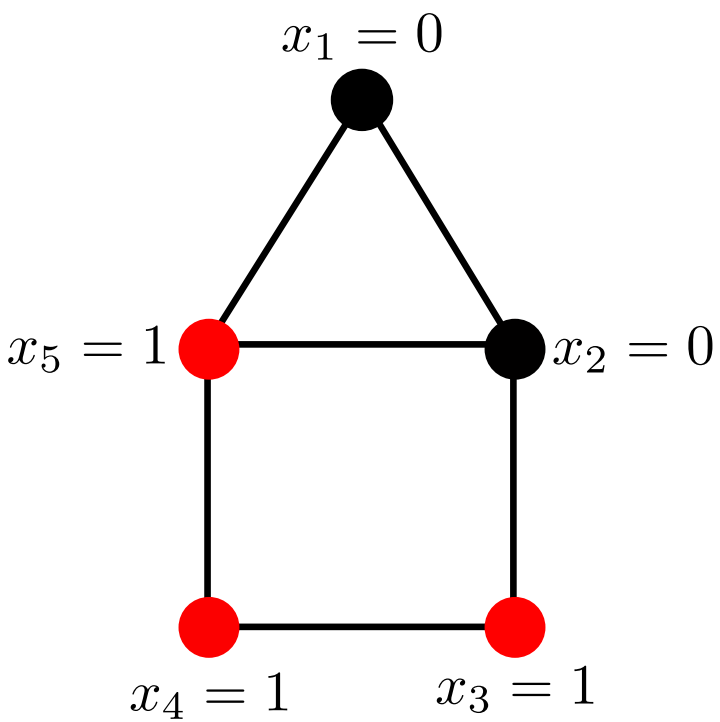
Motivating example: minimum vertex cover

Let's assign coloured vertices to have value 1, and un-coloured 0.



Now that we have our variables, how do we come up with a minimizable cost function that represents the problem?

Motivating example: minimum vertex cover

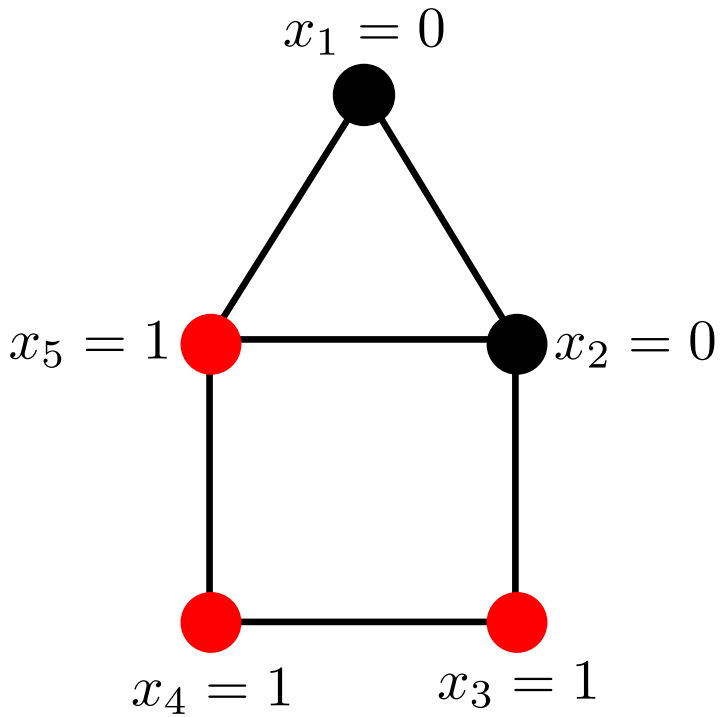


We need every edge to be next to a coloured vertex.

Design a function on two vertices that is minimized when colouring is valid, and maximized if it's not.

$$f(x_i, x_j) = ??$$

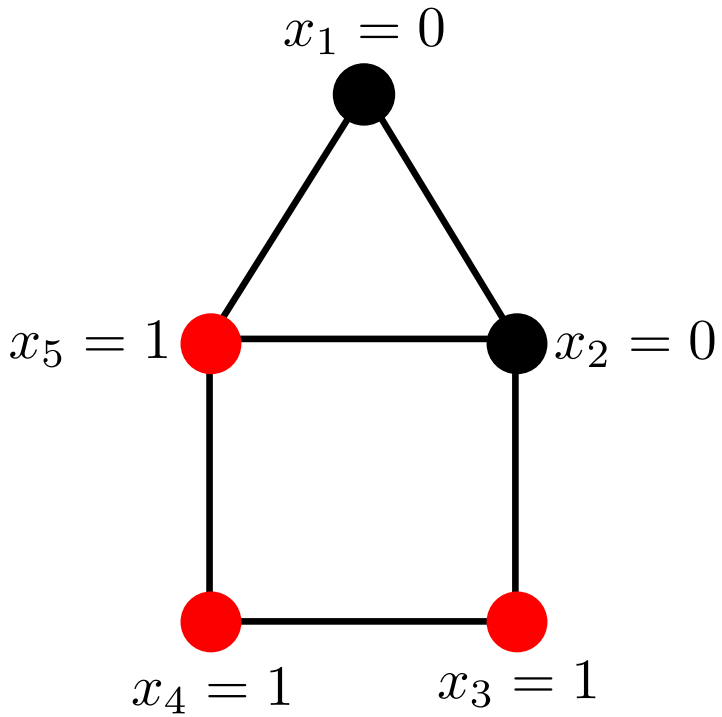
Motivating example: minimum vertex cover



Consider for each edge ij the function

The possible values are:

Motivating example: minimum vertex cover

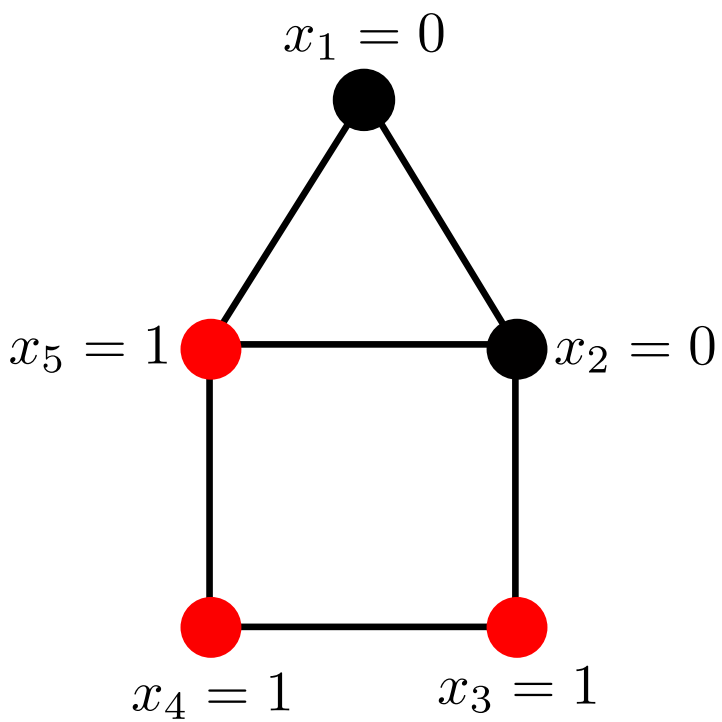


Then in an optimal colouring,

for all edges $ij \in E$.

So we can write

Motivating example: minimum vertex cover



However recall that we also want to colour the fewest vertices. The cost should also depend on the number of coloured vertices.

Solution: add to our cost

Motivating example: minimum vertex cover

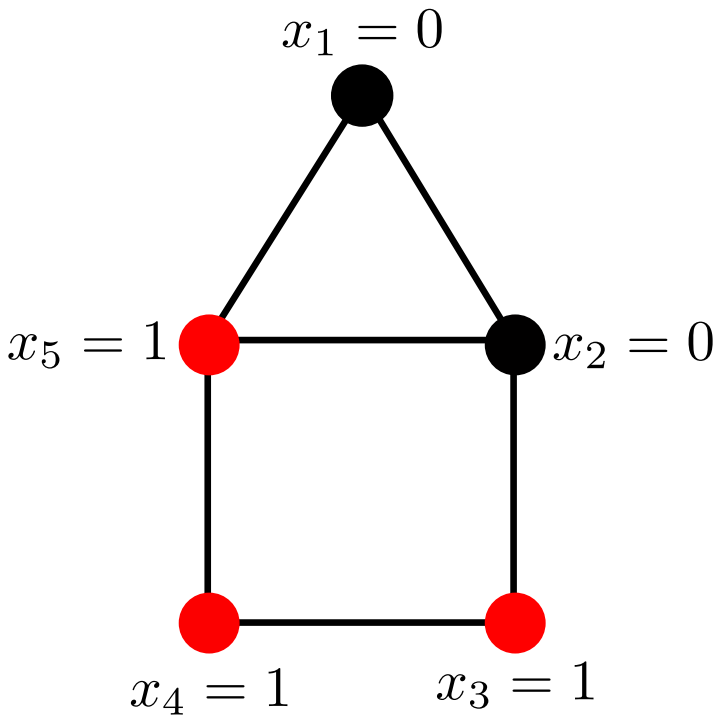
The full cost function is then

Next steps:

1. Turn this into a Hamiltonian over qubits
2. Find its minimum energy

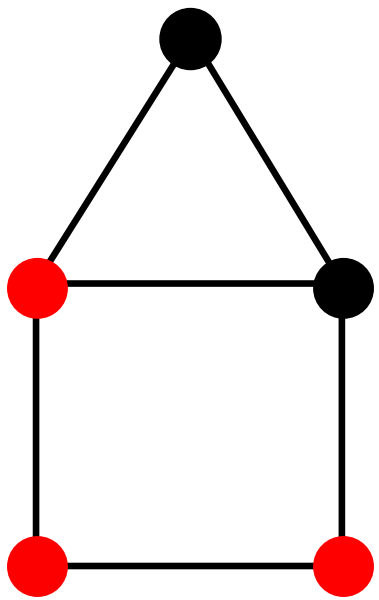
Hamiltonian translation

How should we do a mapping from this to qubits?



Hamiltonian translation

What should we use to represent cost?



Hamiltonian translation

Mathematically, we can make the mapping

This associates

- $x_i = 0$ to $z_i = 1$ (corresponds to $|0\rangle$)
- $x_i = 1$ to $z_i = -1$ (corresponds to $|1\rangle$)

A complete derivation of the cost function is provided as an appendix at the end of the lecture slides.

Hamiltonian translation

We will take the result:

Remember what the z_i represent; how can we express this cost function as a Hamiltonian?

Hamiltonian translation

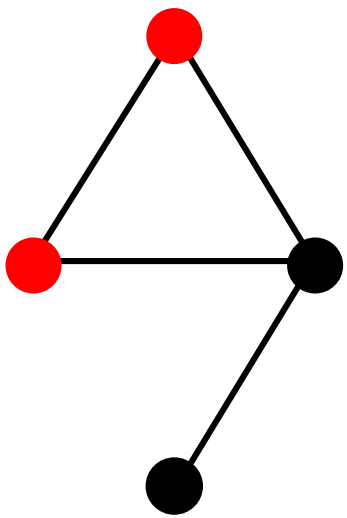
$$\hat{H} = \sum_{ij \in E} (Z_i + Z_j + Z_i Z_j) - 2 \sum_{i \in V} Z_i$$

This makes sense:

Hamiltonian translation

$$\hat{H} = \sum_{ij \in E} (Z_i + Z_j + Z_i Z_j) - 2 \sum_{i \in V} Z_i$$

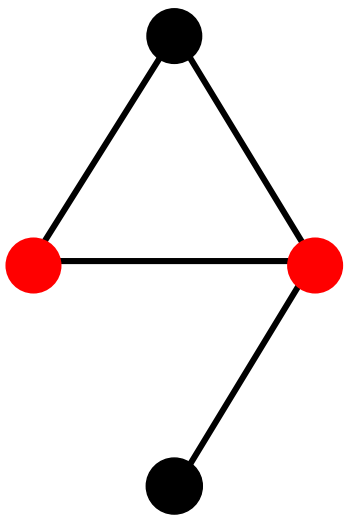
Try it: what is the energy of this *invalid* colouring?



Hamiltonian translation

$$\hat{H} = \sum_{ij \in E} (Z_i + Z_j + Z_i Z_j) - 2 \sum_{i \in V} Z_i$$

Try it: what is the energy of this *valid* colouring?



Next time

Content:

- Solving combinatorial optimization problems with QAOA in PennyLane

Action items:

1. Technical assignment 2
2. Work on prototype implementation for project

Recommended reading (PennyLane demos):

- `https://pennylane.ai/qml/demos/tutorial_qaoa_intro.html`
- `https://pennylane.ai/qml/demos/tutorial_qaoa_maxcut.html`

Full derivation: Hamiltonian translation

In order

We will make the mapping

$$x_i \to \frac{1}{2}(1 - z_i), \quad , z_i \in \{-1, 1\}$$

This associates $x_i = 0$ to $z_i = 1$ (corresponds to $|0\rangle$), and $x_i = 1$ to $z_i = -1$ (corresponds to $|1\rangle$).

Full derivation: Hamiltonian translation

Let's expand our cost function and make this substitution.

$$\sum_{ij \in E} (1 - x_i)(1 - x_j) + \sum_{i \in V} x_i$$

$$\sum_{ij \in E} (1 - x_i - x_j + x_i x_j) + \sum_{i \in V} x_i$$

Full derivation: Hamiltonian translation

$$\sum_{ij \in E} (1 - x_i - x_j + x_i x_j) + \sum_{i \in V} x_i$$

Substitute:

$$\sum_{ij \in E} \left(1 - \frac{1}{2}(1 - z_i) - \frac{1}{2}(1 - z_j) + \frac{1}{4}(1 - z_i)(1 - z_j)\right) + \sum_{i \in V} \frac{1}{2}(1 - z_i)$$

Expand:

$$\sum_{ij \in E} \left(1 - \frac{1}{2} + \frac{1}{2} z_i - \frac{1}{2} + \frac{1}{2} z_j + \frac{1}{4} - \frac{1}{4} z_i - \frac{1}{4} z_j + \frac{1}{4} z_i z_j\right) + \sum_{i \in V} \frac{1}{2}(1 - z_i)$$

Collect:

$$\sum_{ij \in E} \left(\frac{1}{4} + \frac{1}{4} z_i + \frac{1}{4} z_j + \frac{1}{4} z_i z_j\right) + \sum_{i \in V} \frac{1}{2}(1 - z_i)$$

Full derivation: Hamiltonian translation

$$\sum_{ij \in E} \left(\frac{1}{4} + \frac{1}{4} z_i + \frac{1}{4} z_j + \frac{1}{4} z_i z_j \right) + \sum_{i \in V} \frac{1}{2} (1 - z_i)$$

Consider now that: the total number of edges and vertices are constant - they will provide only an "offset" to the cost, and the values of the variables don't matter.

$$\sum_{ij \in E} \left(\frac{1}{4} z_i + \frac{1}{4} z_j + \frac{1}{4} z_i z_j \right) - \sum_{i \in V} \frac{1}{2} z_i$$

And finally, the absolute value doesn't matter, so we can rescale:

$$\sum_{ij \in E} (z_i + z_j + z_i z_j) - 2 \sum_{i \in V} z_i$$

Full derivation: Hamiltonian translation

Can also weight the terms differently depending on which constraint is more important (i.e., if you care more about just getting a valid colouring, weight the first one more).

$$\gamma \sum_{ij \in E} (z_i + z_j + z_i z_j) - 2\lambda \sum_{i \in V} z_i$$

To turn this into a Hamiltonian, recall that

- Each z_i represents an expectation value of Z_i
- Computing expectation values is linear

Full derivation: Hamiltonian translation

$$\gamma \sum_{ij \in E} (z_i + z_j + z_i z_j) - 2\lambda \sum_{i \in V} z_i$$

$$\hat{H} = \gamma \sum_{ij \in E} (Z_i + Z_j + Z_i Z_j) - 2\lambda \sum_{i \in V} Z_i$$

Next time: we will look at the actual QAOA that can find the optimal configuration / minimum energy.