Understanding the Quantum Annealing Process

Quantum Annealing is a quantum optimization method that finds the ground state of a problem Hamiltonian by slowly evolving a quantum system from an initial easy state to the problem state

The system starts in a simple superposition state and transitions to the problem Hamiltonian using a time-dependent schedule, ideally staying in the ground state throughout

Quantum tunneling allows the system to escape deep but narrow local minima by passing through energy barriers instead of climbing over them–this helps search rugged energy landscapes more effectively than classical methods.

In our case, the Max-Cut problem maps naturally onto an Ising Hamiltonian, so quantum annealing can find high-quality cuts by identifying the ground state of that mapped problem

The Max Cut Problem

Unweighted Max-Cut is a graph optimization problem where the goal is to divide the vertices into two sets such that the total weight of edges crossing the partition is maximized.

It is an NP-hard problem, meaning that finding the exact optimal cut becomes computationally expensive for large graphs

Max-Cut can be written as a quadratic Ising Hamiltonian, where each vertex is represented by a spin variable s in {-1, +1} and the cost function becomes an energy expression

The ground state of the corresponding Ising Hamiltonian directly encodes the maximum cut, which makes Max-Cut an ideal candidate for quantum annealing

Hamiltonion and Initialization

→ Defining the Max-Cut Target Hamiltonian and Initial Hamiltonian

- For the target Hamiltonian, we incorporate a ZZ interaction for each edge:
  - $H_p = \sum Z_i Z_j$
  - This configuration yields a net +1 for two vertices within the same set and -1 for those in different sets. The ground state of this Hamiltonian encodes the solution to our Max-Cut problem.
- The initial Hamiltonian is defined as:
  - $H_i = -\sum X_i$
  - which represents a uniform superposition across all 2^n computational basis states.

→ Annealing Schedule

- The annealing schedule is given by:
  - $H(t) = (1-s(t))H_i + s(t)H_p$

    where **$s(t) = (t/T)^2$**.
- Starts with superposition → slowly turns into problem Hamiltonian.

Trotterized Quantum Annealing Circuit

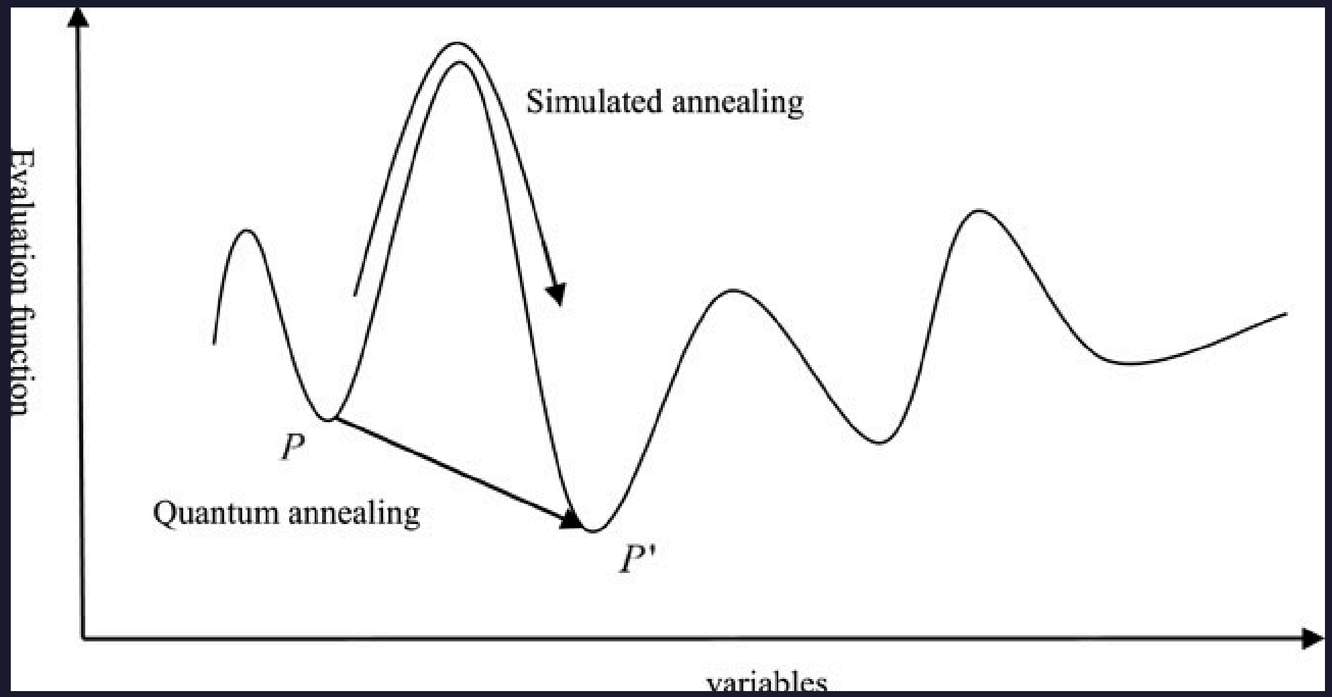
→ Direct implementation of continuous evolution is not possible

- H_i and H_p do not commute.
- An approximation is necessary: $e^{-i(H_i+H_p)\Delta t} \approx e^{-iH_i\Delta t}\, e^{-iH_p\Delta t}$

→ Trotterization with M = 200 steps:

  - Divide the total time T into smaller intervals: dt = T/M.
  - For each interval, apply both driver and problem terms.
- Driver evolution using RX gates:
  - $e^{-i(1-s(t))(Hi)\Delta t}$ translates to:
    - RX(−2(1−s(t))dt)
- Problem evolution through ZZ coupling:
  - $e^{-i(s(t))Z_iZ_j\Delta t}$ is executed using:
    - CNOT → RZ(2s(t)dt) → CNOT
- One Trotter step includes:
  - Applying RX gates on each qubit.
  - ZZ block on each edge

A picture explaining why quantum annealing works better than classical annealing



Sampling, Max-Cut Evaluation, and Results

→ Final measurement

- After Trotter steps, draw bitstrings via qml.sample().
- Each bitstring = one spin configuration.
- Map bits to spins: 0 → +1, 1 → −1
- Max-Cut score per sample: $C(s) = \sum_{(i,j)} (1 - s_i s_j) / 2$

→ Select the winner

- Find the sample with the largest score.
- Show that and score.

→ Plot the graph

- +1 blue, −1 red; show the partition.

Output Example: