@Brendan_In_Byte

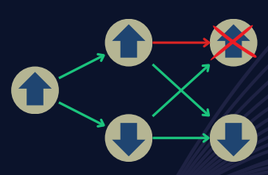
Quantum Algorithms

Algorithms that use quantum phenomena to solve certain problems exponentially faster than classical computers.

$$|\Psi\rangle = a|0\rangle + b|1\rangle$$

Qubits can be in multiple states simultaneously

Quantum computers can perform operations on all qubit states at once → *Potentially outperforms classical computers*



*Quantum computers use **qubits** in superposition, which allow simultaneous evaluation of multiple computational paths. The probability amplitudes of the qubits overlap to provide efficient paths to a solution.*

BUT

Not as simple as just doing everything in parallel:

Need clever algorithm designs to harness this power effectively.

Devising a quantum algorithm

1.Find a problem

where quantum systems have inherent advantages.
These are where problem's solution space grows exponentially with input size. E.g.:

- *Protein folding: Classical MD simulations require $\mathcal{O}(3^n)$ coordinates for N atoms : Quantum advantage candidate*
- *Option pricing: Limited state evolution → Better suited for classical Fourier methods*

2.Problem encoding

Convert the problem into a quantum system.

For example, consider the travelling salesman problem: we want to find the shortest path between N houses. We define a Hamiltonian ("energy") for the system, H, so that

$$H = \min\left(\Sigma_{i,j,k} d_{ik} x_{ij} x_{k,j+1} + A_1 \Sigma_i (1 - \Sigma_j x_{ij})^2 + A_2 \Sigma_j (1 - \Sigma_i x_{ij})^2\right)$$

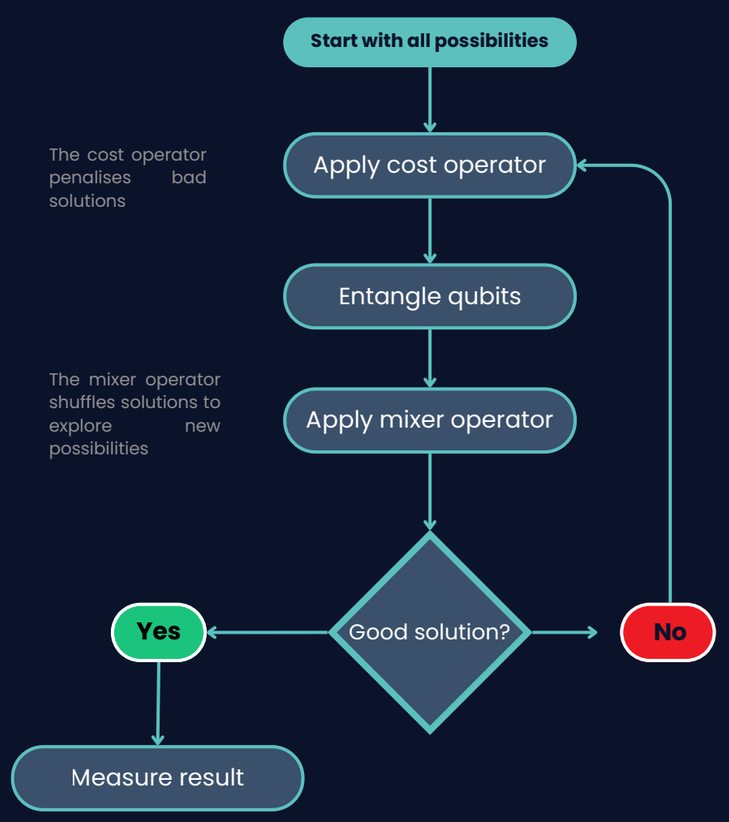
for distances between houses d and binary variable

$$x_{ij} = \begin{cases} 1 & \text{ith house visited in position j of tour} \\ 0 & \text{otherwise} \end{cases}$$

Minimising this Hamiltonian gives the solution to the answer. In a quantum annealer, this can be physically mapped to qubits that have this Hamiltonian, and the system will naturally evolve to find the lowest energy solution which gives the answer. For gate-based systems, we can also use variational quantum algorithms like QAOA.

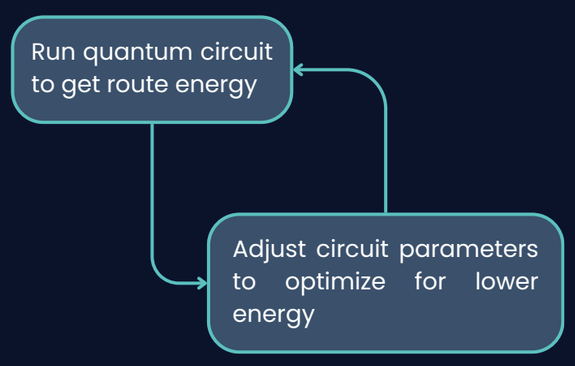
2.1. Circuit construction

Build a quantum recipe using these components:



3.Parameter tuning

Use classical computers to optimize quantum steps.



Key techniques for quantum algorithms

A non-exhaustive list

Quantum walks

Quantum analogs of classical random walks, leveraging principles of quantum mechanics such as superposition, interference, and unitary evolution. Two types exist: discrete-time and continuous-time walks. Discrete-time walks use shift operators while continuous-time walks use graph Laplacians. Used in Grover's search, element distinctness, etc.

Quantum Fourier Transform

Quantum analogue of the classical discrete Fourier transform (DFT)

$$|x\rangle \to \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\pi i x y}{N}} |y\rangle$$

Useful in Shor's algorithm for factoring, quantum phase estimation, and solving the hidden subgroup problem. Can be implemented through Hadamard gates and controlled phase gates.

Amplitude amplification

Technique that amplifies the probability of measuring "good" outcomes in a quantum state, providing a quadratic speedup over many classical algorithms. Useful in Grover's Search Algorithm, error reduction, fault detection, optimization, etc.

Phase estimation

Algorithm used to estimate the phase associated with an eigenvalue of a unitary operator. Used in Shor's algorithm, quantum simulation, linear systems of equations, quantum counting, etc.

Hybrid quantum-classical algorithms

Algorithms that combine the strengths of quantum and classical computing to tackle complex computational problems more efficiently, leveraging quantum systems for specific tasks while utilizing classical computers for other parts of the computation. Notable examples include VQEs, QAOAs, etc.

Warning: Math

Examples of quantum algorithms

Shor's algorithm

Meant for integer factorization, with applications in cryptography e.g. breaking RSA encryption

Given a large integer N to be factorized, the steps are as follows:

Classical preprocessing

Select a random integer a such that 1 < a < N → Compute the greatest common divisor (GCD) of a and N using Euclid's algorithm. If GCD(a,N) > 1, then N is already factored. Otherwise...

Goal: **find period r** of a modular function $f(x) = a^x \mod N$

Where the period r is the smallest positive integer such that $a^r \bmod N = 1$

f(x) is periodic with some period r, i.e. f(x + r) = f(x) because of how the modulo operation works. Every period r tells information about the "size" of the prime factors of N.

Quantum shenanigans

Superposition: prepare a quantum register in a superposition of all possible values of x $|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$ *Sets problem up to take advantage of superposition*

Modular exponentiation: apply a quantum circuit that computes $f(x) = a^x \mod N$
This entangles the input register with an output register $|x\rangle|0\rangle \to |x\rangle|f(x)\rangle$

This entanglement produces a quantum state where each input x is uniquely paired with its corresponding output f(x), allowing the QFT to extract information while evaluating all possible values of f(x) simultaneously.

Quantum Fourier Transform (QFT): perform the QFT on the input register to extract information about the periodicity of f(x).

Measurement: Measure the quantum state, yielding an integer related to r.

Classical postprocessing

Once the period r is found:

1. Check if r satisfies certain conditions (e.g., r must be even - if not, then another integer a is chosen.)
2. Compute $GCD(a^{\frac{r}{2}} - 1, N)\ GCD(a^{\frac{r}{2}} + 1, N)$

One of these will yield a non-trivial factor of N.

Because $a^r - 1 = (a^{\frac{r}{2}} - 1)(a^{\frac{r}{2}} + 1)$ *and since we require* $a^r = 1 \bmod N$ *Thus* $(a^{\frac{r}{2}} - 1)(a^{\frac{r}{2}} + 1) = 0 \bmod N$

Grover's algorithm

Meant for searching unstructured databases or solving Boolean problems. Specifically, solving the problem of finding an input x_0 such that

$f(x_0) = 1$ for a given function $f(x) : \{0,1\}^n \to \{0,1\}$

Which is equivalent to searching for an item in an unsorted database of size $N = 2^n$

Start with an n-qubit register and apply Hadamard gates to create superposition over all possible states $|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$ *Represents all potential solutions simultaneously*

Oracle operator marks the correct solution by flipping its phase:
$O_f|x\rangle = \begin{cases} -|x\rangle & \text{if f(x)} = 1 \\ |x\rangle & \text{if f(x)} = 0 \end{cases}$

Diffusion operator amplifies the marked state amplitude while reducing the amplitude of others
$D = 2|\psi\rangle\langle\psi| - I$ *"-I" term selectively boosts correct solutions*

After around $\mathcal{O}(\sqrt{N})$ iterations

Measure the final state to yield the marked solution with high probability.

Iterative ID + signal boosting for the correct solution. Each iteration slowly filters out the states not close to the answer.

@Brendan_In_Byte