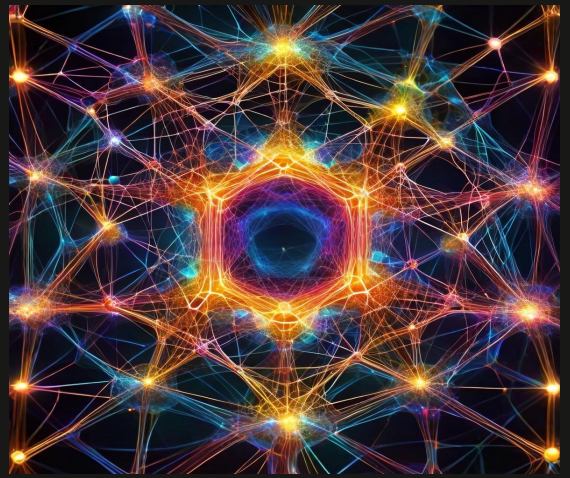
Quantum Algorithms on Graphs

Asaf Ferber, UC, Irvine.

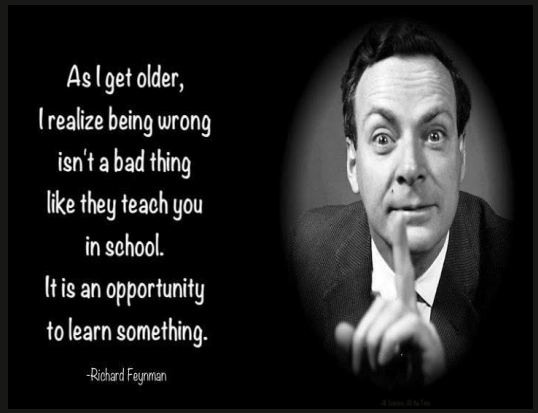


*Research is partially supported by AFOSR.

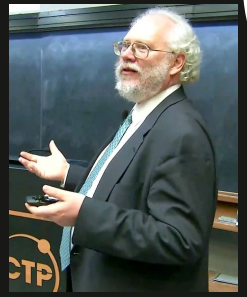
General Introduction to Quantum Computing

Quantum computing is a transformative field that processes information based on the principles of **quantum mechanics**, such as *superposition* and *entanglement*. Unlike classical bits, which represent either a 0 or a 1, **qubits** can exist in multiple states simultaneously, enabling quantum computers to perform many calculations in parallel.

The field's origins trace back to the 1980s, with foundational contributions from pioneers like **Richard Feynman** and **David Deutsch**, who recognized that classical computers struggle to simulate quantum systems. Feynman's vision was to use quantum computers to model quantum phenomena efficiently, while Deutsch proposed the idea of a universal quantum computer, capable of solving problems beyond the reach of classical machines.



General Introduction to Quantum Computing



Quantum computing has the potential to revolutionize multiple fields:

- **Cryptography:** Quantum algorithms, such as **Shor's algorithm**, can factor large numbers exponentially faster than classical methods, threatening the security of widely used encryption protocols like RSA.
- **Optimization:** Quantum computers can accelerate solving complex optimization problems, crucial in industries like logistics, finance, and artificial intelligence, with algorithms such as **quantum annealing** or **Grover's algorithm** for searching unsorted databases.
- **Material Science and Chemistry:** Quantum computers can simulate molecules and materials with high precision, offering insights that could lead to breakthroughs in drug discovery, energy production, and materials engineering.

As this technology evolves, **quantum algorithms on graphs** are emerging as a critical area of research, with potential applications in network theory, combinatorics, and beyond.

General Introduction to Quantum Computing

Classical computers (PCs, HPCs, laptops, etc.) are limited by **locality** and the classical fact that systems exist in only one state at a time.

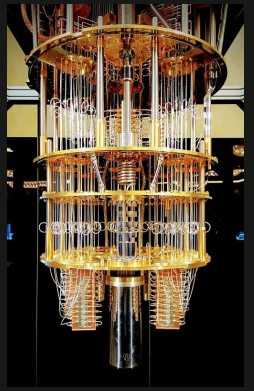
Quantum systems differ:

- They can exist in a **superposition** of states and show **interference** effects.
- **Entanglement** allows spatially separated systems to be connected, enabling non-local effects.

Quantum computation merges two key scientific strands:

- **Quantum mechanics** (Planck, Einstein, Bohr, Schrödinger, Heisenberg, 1900–1925)
- **Computer science** (Turing, 1936)

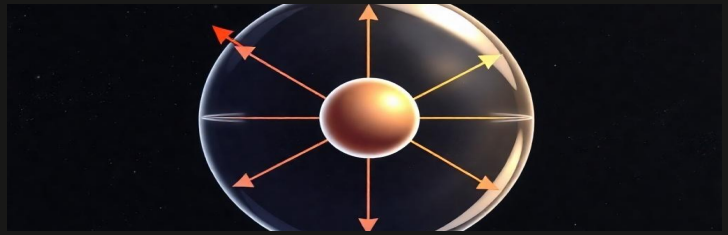
The goal is to discover **quantum algorithms** that outperform classical algorithms.



Key Historical Milestones in Quantum Computation

- **1980s: Early Foundations: Yuri Manin**, **Richard Feynman**, and **Paul Benioff** proposed the idea of quantum computation as a means to simulate physical systems.
- **1985: Universal Quantum Computer: David Deutsch** defined the **universal quantum Turing machine**, the theoretical framework for quantum computation.
- **1994: Shor's Algorithm: Peter Shor** developed an efficient quantum algorithm for **integer factorization**, posing a threat to classical cryptography (e.g., RSA).
- **1996: Grover's Algorithm: Lov Grover** introduced an algorithm that provides a quadratic speed-up for **searching unsorted databases**, reducing the time from O(N) to O(√N) compared to classical search algorithms.
- **1984 Quantum Cryptography:** The unbreakable **quantum cryptography protocol** was developed by **Bennett** and **Brassard** (BB84), ensuring secure communication even in a quantum era.

Mathematical Introduction to Quantum Computing



What is a Qubit?

A qubit is the basic unit of quantum information, analogous to a bit in classical computing. Unlike a bit, which can be 0 or 1, a qubit can exist in a superposition of both states simultaneously.

Formally, a qubit is just a unit vector in $\mathbb{C}^2$.

We use the Dirac's `bra-ket' notation and write $|\phi\rangle$ for a qubit. Throughout, we fix some orthonormal basis for $\mathbb{C}^2$ and denote its vectors by $|0\rangle$ and $|1\rangle$

Operations on Qubits

Quantum Gates: Analogous to classical logic gates, quantum gates manipulate qubits. Here a quantum gate is an invertible linear transformation.

Examples include the Hadamard gate, Pauli-X gate, and CNOT gate.

Measurement: Observing a qubit collapses its superposition to a definite state, either 0 or 1.

Formally, if we let $|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$, then the qubit collapses to $|0\rangle$ with probability α^2 and $|1\rangle$ o/w.


SCHRÖDINGER PLATES
THEY'RE BOTH BROKEN AND NOT BROKEN

Mathematical Introduction to Quantum Computing

Wait! Can a single qubit store an **infinite amount of information**? What stops us from creating **clones** and measuring the qubit repeatedly to learn its amplitudes?

Theorem (No Cloning): An unknown qubit cannot be cloned.

This means that if we have a qubit in an unknown state, we cannot clone it to obtain multiple copies for measurement.

* The No-Cloning Theorem is a fundamental principle that underlies many aspects of quantum information theory, including **quantum computing**, **quantum cryptography**, and the secure transmission of quantum states.

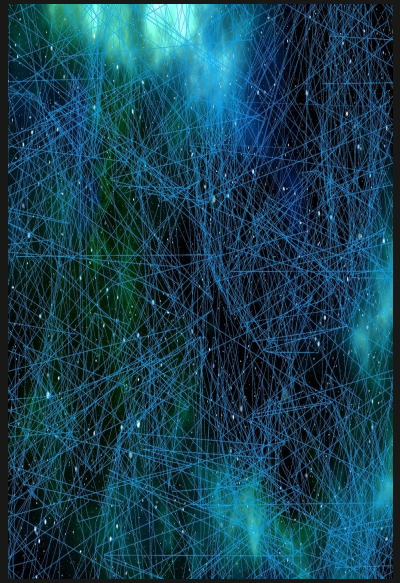
Mathematical Introduction to Quantum Computing

1 qubit gates:

1. **Hadamard gate (H):** Is defined by: $H|0\rangle = 1/\sqrt{2}\,(|0\rangle + |1\rangle)$, $H|1\rangle = 1/\sqrt{2}\,(|0\rangle - |1\rangle)$.
2. **Pauli-X Gate (X):** The quantum equivalent of the classical **NOT gate**, it flips the state of a qubit.
3. **Pauli-Z Gate (Z):** Applies a phase shift of π to the $|1\rangle$ state, leaving $|0\rangle$ unchanged. It's a **phase-flip gate**.
4. **Pauli-Y Gate (Y):** Defined by $Y = iXZ$.

A 2 qubit gate:

CNOT Gate (Controlled-NOT): A **two-qubit gate** where the second qubit is flipped if the first qubit (control) is $|1\rangle$. It's essential for creating **entanglement**.



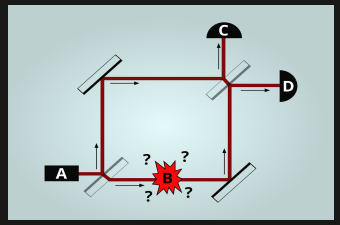
Mathematical Introduction to Quantum Computing

How is quantum computation useful?

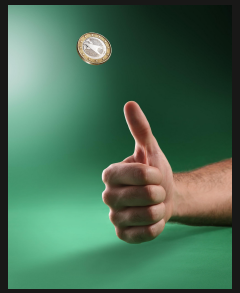
We highlight two advantages of the quantum setting over the classical setting even by using one qubit:

1. **Memory Advantage: Bias Detection with One Qubit** Suppose we are given a coin that can be either **unbiased** or **ε-biased**. Using quantum computation, even with just **one qubit**, we can determine which one it is with greater efficiency than classical methods.

2. **Impossibility Result:** The **Elitzur-Vaidman Bomb Tester** A thought experiment introduced in 1993 by Israeli physicists **Avshalom Elitzur** and **Lev Vaidman**. This paradoxical quantum phenomenon demonstrates the concept of **interaction-free measurement**, allowing us to gain information about an object without directly interacting with it—something impossible in the classical setting.



Example: Epsilon-Biased Coin Decision

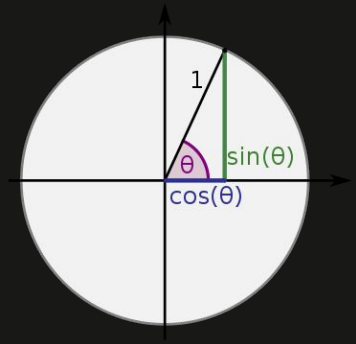


Classical Approach

To determine if a coin is ε-biased, one would flip the coin multiple times and analyze the results statistically. By Chernoff's, one should flip the coin around $1/\epsilon^2$ trials, and therefore, we need at least log(1/ε) bits for storage.

Quantum Approach

With Quantum, we can solve this problem by using a single qubit!



Example: The Elitzur-Vaidman Bomb

The Classical Dilemma:

- You have a bomb, but you don’t know whether it is **functional** or a **dud**.
- Classically, the only way to check would involve triggering the detector, which would cause the bomb to explode if it’s functional.

The Quantum Solution:

- Using a **quantum detector**, it’s possible to design a procedure that can determine if the bomb is **functional** or a **dud**.
- This method allows us to detect a functional bomb with **high probability**, while keeping the chance of triggering an explosion **very low**.

Interactions of Qubits

When we have **multiple qubits**, their combined state lives in the **tensor product** space.

For two qubits, their joint state is written as:

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

In general, the combined state of n qubits lives in a 2^n - dimensional space and can be written as:

$$\sum_{x=0}^{2^n-1} \alpha_x |x\rangle$$

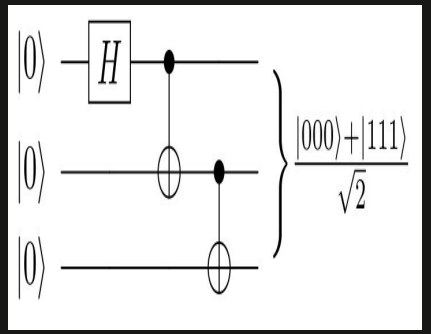
States that cannot be written as tensor products (e.g., that cannot be written as $(\alpha|0\rangle + \beta|1\rangle) \otimes (\gamma|0\rangle + \delta|1\rangle)$) are called **entangled.**

Interactions of Qubits – Entanglement

Entanglement is a unique quantum phenomenon where the states of two qubits become intertwined. Measuring the state of one qubit instantly influences the state of the other, regardless of the distance between them.

This property is crucial for quantum computing, enabling complex computations and secure communication.

We now show how to entangle two qubits



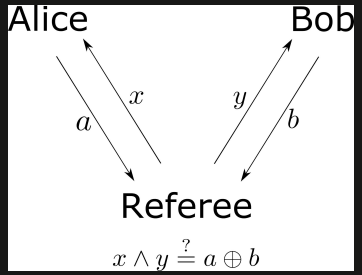
This process involves applying quantum gates as follows:

1. Take two qubits, each initialized to $|0\rangle$. Their combined state is: $|00\rangle = |0\rangle \otimes |0\rangle$.
2. Apply a Hadamard Gate to the First Qubit. This transforms the combined state to $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |0\rangle$
3. Finally, apply the CNOT gate to the 2nd qubit, to obtain the ***Bell state** $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$

*Also known as EPR-pairs, in honor of Einstein, Podolsky, and Rosen, who examined such states and their seemingly paradoxical properties.

An application using Entanglement

- **Scenario:**
  - **Alice** and **Bob** want to convince an adversary, **Charlie**, that a cycle of length 2n+1 is **2-colorable**.
  - They can agree on a strategy beforehand, but **cannot communicate** after the game begins.
- **The Rules:**
  - In each step, **Charlie** gives **Alice** and **Bob** two vertices, **a** and **b**, which can be adjacent or the same.
  - **Alice** and **Bob** must assign either **Red** or **Blue** to their vertex.
  - If a=b, they must return the **same color**; if a≠b, they must return **different colors**.
- **Classical Outcome:**
  - Using classical methods, they can win with probability 1−1/(2n+1).
- **Quantum Advantage (Entanglement):**
  - By using **quantum entanglement**, they can increase their winning probability to $1-\frac{1}{n^2}$, a significant improvement over the classical approach.



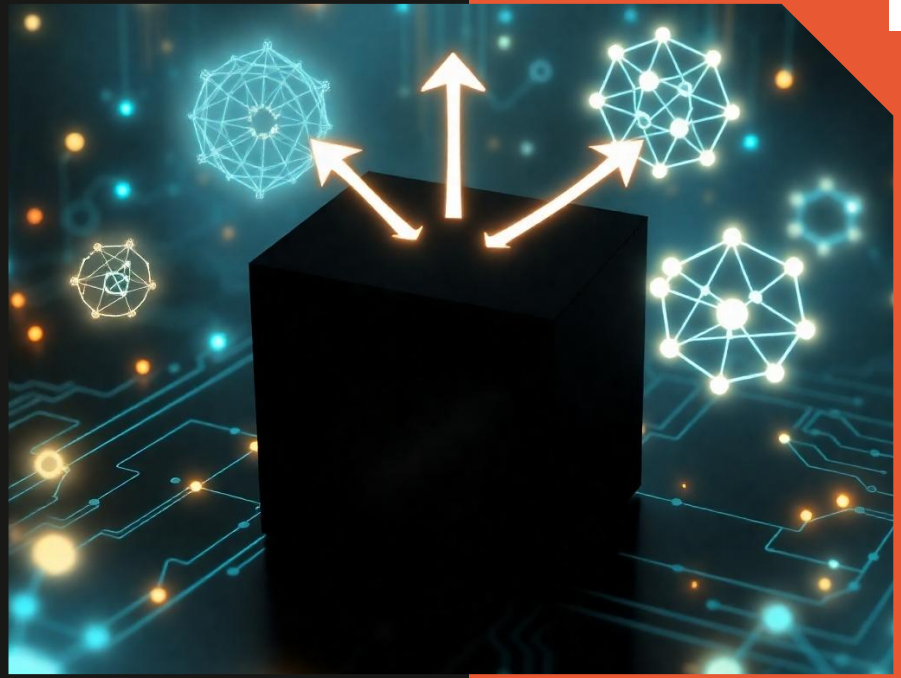
The Black Box Model

The **Black Box Model** (or **Oracle Model**) is a framework where we treat a function as a "black box" or an **oracle**.

The inner workings of the function are **unknown** to us, but we can query the oracle to get output for given inputs.

Classical Black Box: Given an input, the oracle returns an output by computing the function classically. Each query provides only one piece of information at a time.

Quantum Black Box (Oracle): Quantum computers allow querying the oracle on a **superposition** of inputs.This leads to more efficient algorithms, as quantum algorithms can extract more information with fewer queries.



The Black Box Model (cont'd)

The model

- Given any function $f : \{0,1\}^n \to \{0,1\}^m$, define the unitary operator O_f by $O_f|x\rangle|y\rangle := |x\rangle|y \oplus f(x)\rangle$

- Now, a **Quantum Query** consists on an application of O_f on any given qubit.
- When m=1, it is sometimes more convenient to work with $O_f|x\rangle := (-1)^{f(x)}|x\rangle$

  * This can be achieved by applying O_f to the state $|x\rangle(|0\rangle - |1\rangle)$ and then measuring the second qubit with respect to the orthonormal basis |+⟩, |-⟩.

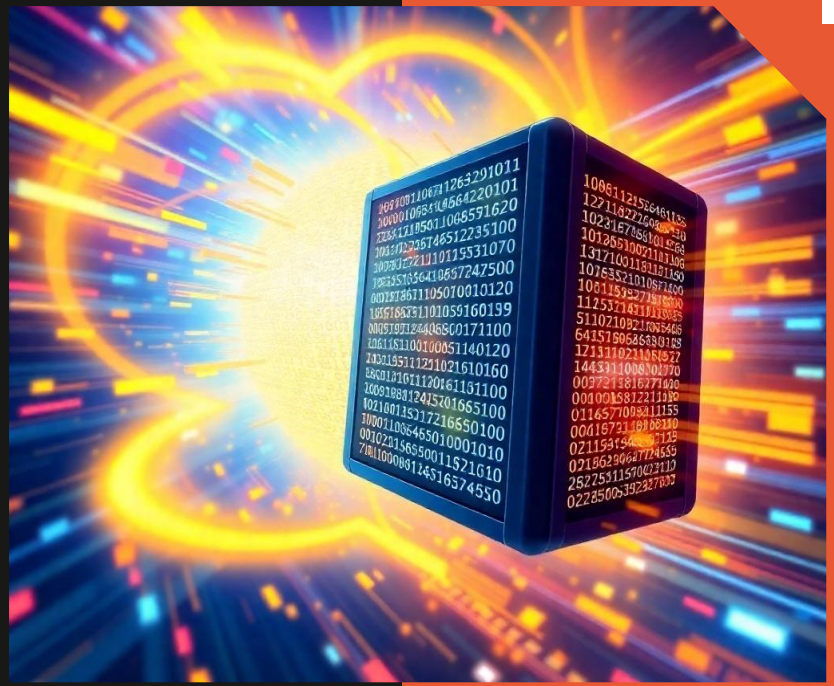
- Can be applied in a superposition. For example, if we apply the operator on the state $|\Psi\rangle := \frac{1}{\sqrt{N}}\sum_x |x\rangle$, we obtain $O_f|\Psi\rangle = \frac{1}{\sqrt{N}}\sum_x (-1)^{f(x)}|x\rangle$.

Example: Grover's Algorithm

Grover's algorithm is a quantum search algorithm that provides a quadratic speedup over classical search algorithms. It can find an item in an unsorted database of N items in O(√N) time, demonstrating the computational efficiency of quantum algorithms.

Proposed by **Lov Grover** in **1996**, this algorithm revolutionized quantum search, showing that quantum computers could achieve a significant speedup over classical search algorithms.

It was one of the earliest algorithms to demonstrate the power of **quantum computation** beyond the classical capabilities.



Grover's Algorithm (cont'd)

Problem Definition

Given a function $f : \{0,1\}^n \to \{0,1\}$, where f(x)=1 for a unique target element x=s and f(x)=0 for all other x.

Goal

Find s with **high probability** using quantum operations.

Oracle Query

Use a quantum **oracle** O_f that marks the solution by flipping the phase of the target state:

$$O_f|x\rangle = \begin{cases} -|x\rangle, & \text{if } x = s, \\ |x\rangle, & \text{otherwise.} \end{cases}$$

The algorithm:

- Start by creating a uniform superposition of all $N = 2^n$ possible inputs:

$$|\psi\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

- **Amplitude Amplification:** Apply **Grover's Diffusion Operator** (inversion about the mean) to amplify the probability of measuring the correct solution: $D = 2|\psi\rangle\langle\psi| - I$

  This operator increases the amplitude of the marked state.

- **Repeat** the oracle and diffusion steps for approximately $\frac{\pi}{4}\sqrt{N}$ iterations to maximize the probability of observing the correct answer.
- **Measure.**

Research Focus: Quantum Algorithms on Graphs

My research explores the application of quantum algorithms to graph theory. This emerging field has the potential to solve complex problems in network analysis, optimization, and more. By leveraging quantum properties like superposition and entanglement, we can develop new algorithms that outperform classical counterparts.

Project 1: Learning a Hidden Graph

Motivating Problem

Imagine a lab scenario with a **collection of chemicals**, where some pairs of chemicals react with one another.

You can mix any **subset of chemicals** in a test tube, and a reaction occurs if and only if at least one **reacting pair** is present.

Goal: Determine all reacting pairs using as few experiments as possible.

Formal Description

Represent the set of chemicals as **vertices** in a graph $G=(V,E)$, where each reacting pair corresponds to an **edge.**

Each experiment corresponds to selecting a subset $S \subseteq V$ and querying whether there is at least one **edge** (a reaction) in the subgraph induced by S.

This type of experiment is called an **OR query**, and our goal is to learn E using as few OR queries as possible.

Project 1: Learning a Hidden Graph

Classic algorithm

Information theoretic lower bound: In each OR query we get 1 bit of information. Since there are $\binom{\binom{n}{2}}{m} = 2^{\Omega\left(m \log(n^2/m)\right)}$ graphs G on n vertices with m edges, the number of OR queries required to learn such a G is of order $m \log(n^2/m)$

Angluin and Chen (2008) gave a simple and optimal algorithm that uses $O(m \log(n^2/m))$ queries.

A quantum version

Ambainis and Montanaro (2014) gave a lower bound of Ω(√m).

In 2021 **Montanaro and Shao** proved:

1. Slightly improved the classical bound to $O(m \log(\sqrt{m} \log n))$.
2. Under the assumptions that G has maximum degree d and is O(1) colorable, they significantly improved the classical bound and obtained $O\left(d^2 m^{\frac{3}{4}} \sqrt{\log n} (\log m)\right)$.

*In particular, for perfect matchings or Hamilton cycles, they obtained a bound of the form $m^{\frac{3}{4}} \log^c n$

Project 1: Learning a Hidden Graph

Our results (F. and Liam Hardiman 2024+)

- Perfect matchings and Hamilton cycles*:

$$\tilde{O}\left(\sqrt{n}\right)$$

- Graphs with maximum degree d and m edges:

$$\tilde{O}(d\sqrt{m})$$

*Up to logarithmic factors, our results match Ambainis and Montanaro's lower bound.

Project 2: Vertex Coloring of a Graph

Problem description

A **proper k-coloring of G** is a coloring of V(G) with k colors such that no edge is monochromatic.

The smallest positive k such that G admits a proper k-coloring is called the **chromatic number** of G, denoted by $\chi(G)$.

While determining $\chi(G)$ is NP-hard (Karp 1972), it is easy to color a graph G with **maximum degree** D using D+1 colors.

A simple **greedy algorithm** achieves this by assigning each vertex a color different from those of its neighbors. Since each vertex has at most D neighbors, there is always at least one available color to ensure no edge becomes monochromatic.

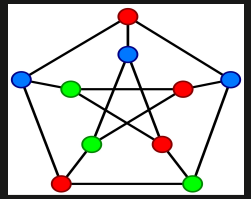
*Without making further assumptions about G, one cannot do with fewer than D+1 colors: consider a clique or an odd cycle.

Project 2: Vertex Coloring of a Graph

The Model

Suppose we have access to an oracle that can answer the following types of queries:

- **Adjacency queries:** Given two vertices u and v, we can ask whether they are adjacent.
- **Neighborhood queries:** For any vertex v and any integer j, we may query the j-th neighbor of v.



Here we judge an algorithm's efficiency by the **number of queries** it makes to such an oracle.

General Framework

Given two algorithms to vertex-color G with c colors:

- **Algorithm 1:** Uses f(D) adjacency queries, where f is monotone decreasing.
- **Algorithm 2:** Makes g(D) neighborhood queries, where g is increasing.

By applying the first algorithm when $D \geq D'$ and the second when $D \leq D'$ we obtain a worst-case query complexity of $f(D') = g(D')$ (this is how we choose D').

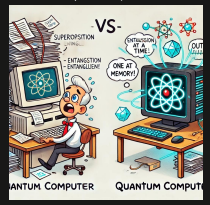
Project 2: Vertex Coloring of a Graph

Performance (classical)

- **Greedy** makes O(|E|) neighborhood queries.
- **Assadi, Chen, and Khanna** (2019) introduced the **palette sparsification** technique and produced a (D+1)-coloring with high probability in $O(n^2 \log^2 n/\Delta)$ adjacency queries.

By combining their algorithm (as f) with greedy (as g), we achieve a (D+1)-coloring in $O(n^{3/2} \log n)$ queries*.

- **Morris and Song** (2021) gave a simple algorithm to (1+ε)D-color G in f(D)= $O(\varepsilon^{-1} n^2/D)$ queries in expectation.



*In the same paper they also showed that this is tight up to the log factor.

Performance (quantum)

Morris and Song: using Grover's algorithm in every step of their algorithm, they obtain a (1+ε)D-coloring within f(D)=$O(n^{3/2} \log n/\sqrt{D})$ queries. Combining it with g(D)=Dn (greedy), they obtain a running time $\tilde{O}(n^{4/3})$.

They also gave a **lower bound** of $\Omega(n)$.

Project 2: Vertex Coloring of a Graph

Our results (Chen, F., and Hardiman 2014+)

- Simple algorithm to (D+1)-color within $O(n^2 \log n/D)$ adjacency queries in expectation.
- Quantum alg. to (D+1)-color within $O(n^{3/2} \log n/\sqrt{D})$ quantum adjacency queries.
- Quantum alg. to (1+ε)D-color within $O(\epsilon^{-2} n \log^2 n\sqrt{D})$ quantum neighborhood queries.

By combining the bound in the last bullet (as g) with the bound in the second (as f), we obtain a running time $\tilde{O}(n^{5/4})$.

Project 2: Vertex Coloring of a Graph

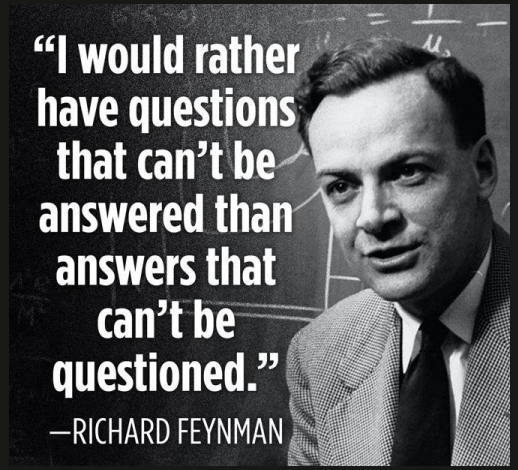
Motivating example

Let G be a random graph on V=[n] (with edge probability p), and fix a partition of V into T intervals. With high probability, the degrees of all vertices are concentrated within an interval around np/T. One can then apply the Greedy algorithm to each interval, making neighborhood queries only within the ``correct'' range for each interval. By using disjoint color palettes for each interval, this approach achieves a (1+o(1))D coloring within n^2p/T queries.

For concentration of the degrees, we need T=np/log n so this gives a O(n log n) algorithm.

More problems

- Improve the bound.
- (D+1)-coloring?
- What if we assume G is triangle free?
- Etc.

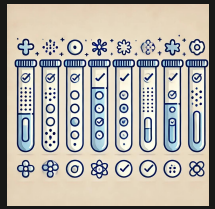


Project 1: Some Proof Ingredients

Combinatorial Group Testing (CGT)

- **Goal**: Efficiently identify a small number of "defective" items within a large set using the minimum number of tests.
- **Applications**: Medical testing, network security, fault detection in manufacturing, data compression.

Group Testing Concept: Instead of testing each item individually, test groups of items:



- If **Positive Result**: At least one defective item in the group.
- If **Negative Result**: All items in the group are non-defective.

It is easy to come up with an algorithm to identify all k defective items within a set of size n within k log(n) OR queries.

Belovs showed in 2014 that one can solve the CGT problem using a quantum algorithm within $O(\sqrt{k})$ quantum OR queries (and this is tight).

Project 1: Some Proof Ingredients

We use this algorithm for CGT to develop a randomized algorithm for finding edges that

cross between independent sets. Our algorithm is inspired by that of Montanaro and Shao, itself inspired by Angluin and Chen's classical algorithm.

Key Lemma (F. and Hardiman)

Let G be a graph on n vertices with maximum degree d. Suppose A and B are two disjoint, non-empty independent sets of vertices in G, and that there are m edges between A and B.

Then there is a quantum algorithm that identifies the m crossing edges in $O(d\sqrt{m}\log n)$ queries.

Project 1: Some Proof Ingredients

Proof (Lemma)

For any subset $T \subseteq B$, querying sets of the form SUT, where S ranges over the subsets of A, reduces the problem of learning $N(T)$ to the CGT problem.

While Belovs' allows us to learn $N(T)$ in $O(\sqrt{|N(T)|})$ queries, we do not learn the individual adjacencies.

To this end: Take $N = 60\ d \log n$ random subsets T of B, where each individual element $b \in B$ appears in T with probability $p=1/(3d)$ independently for all T.

By Chernoff and a union bound, whp each $b \in B$ appears in $C \log n$ such sets T.

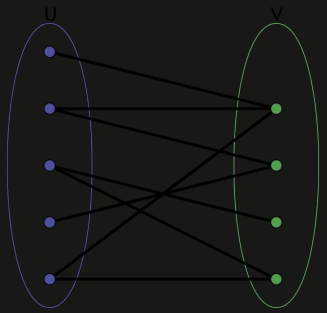
Finally, we show that whp we have that for each $a \in A$ and $b \in B$, we have:

For all T, if $b \in T$ then $a \in N(T)$ $\iff$ ab is an edge.

This completes the proof. □

Project 1: Some Proof Ingredients

Random Partitioning (say for matchings)



Partition the graph into √n random sets.

Pair them up, and learn the edges between each pair.

Then, merge pairs, pair the merged sets, and REPEAT.

Why does it work?

In each step j, every set is a random and of size $2^j\sqrt{n}$.

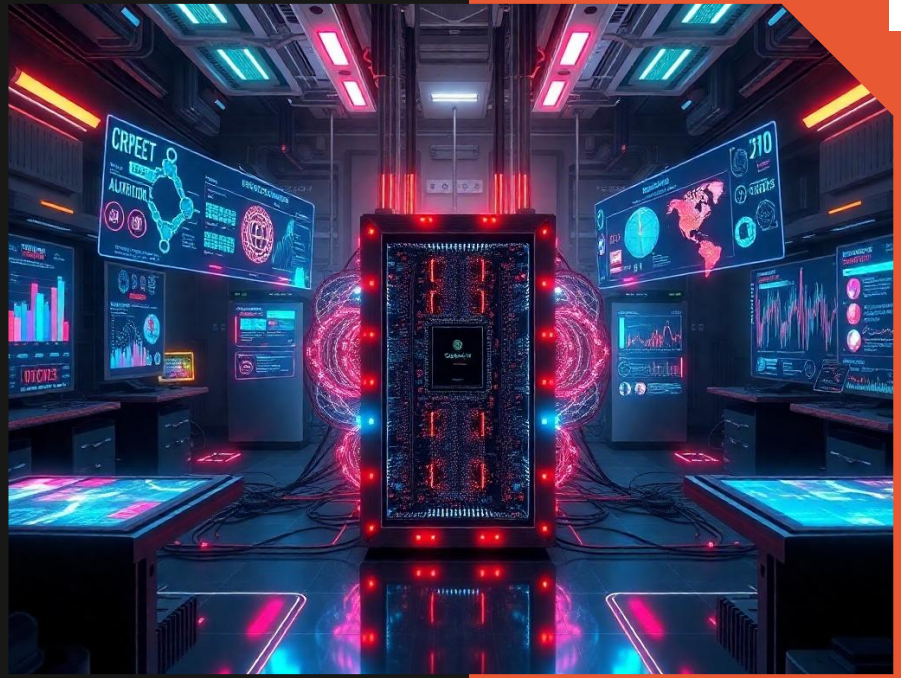
The expected number of edges between every pair is 2^{2j}, and it takes around 2^j queries to learn it.

Since there are $\frac{\sqrt{n}}{2^j}$ pairs, and since there are at most $\log n$ steps, the entire procedure takes at most $\sum_{j=0}^{\log n} \frac{\sqrt{n}}{2^j} \cdot 2^j = O(\sqrt{n}\log n)$ Queries to learn the graph.

Conclusion and Future Directions

Quantum computing is a rapidly evolving field with immense potential. As scientists continue to develop new algorithms and technologies, the possibilities are endless. Future research will focus on practical implementations and expanding the applications of quantum computing.

Some potential subareas of research: Quantum error correcting codes, non-local games (e.g., quantum chromatic number of a graph), and more.



Thank you for your attention, and I welcome any questions or discussions.