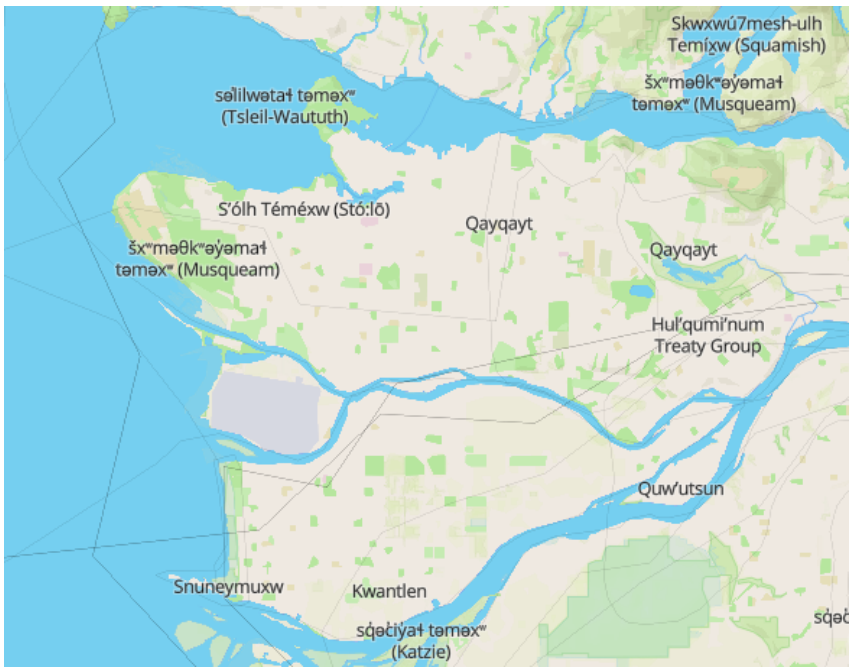
CPEN 416 / EECE 571O

Lecture 01: Overview and intro to gate model quantum computing

Thursday 04 September 2025

Land acknowledgement

The land on which we gather today is the traditional, ancestral, and unceded territory of the Musqueam People.



Explore more:

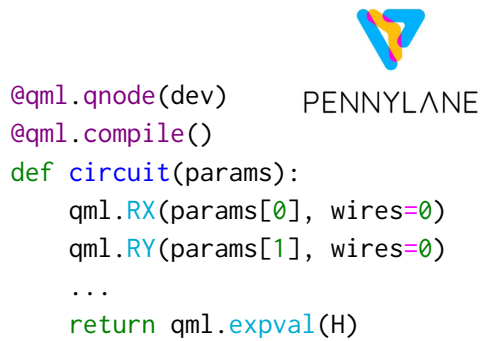
- `https://www.musqueam.bc.ca/`
- `https://native-land.ca/`

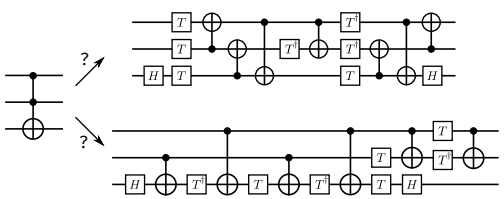
Intros

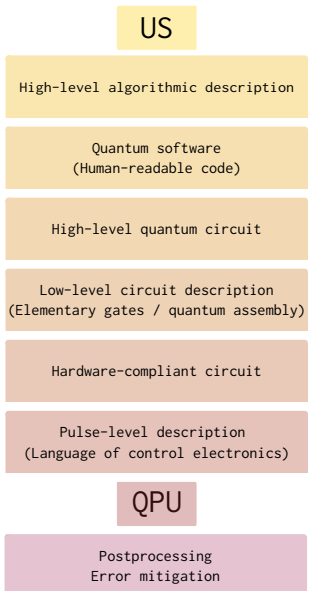
Olivia Di Matteo – olivia@ece.ubc.ca

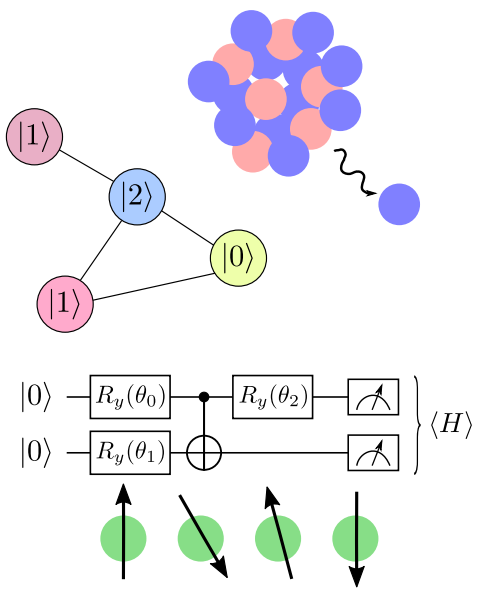
I have a physics background. My group works on open-source quantum software and algorithms.

>QSAR QUANTUM SOFTWARE & ALGORITHMS RESEARCH









Billion-dollar question

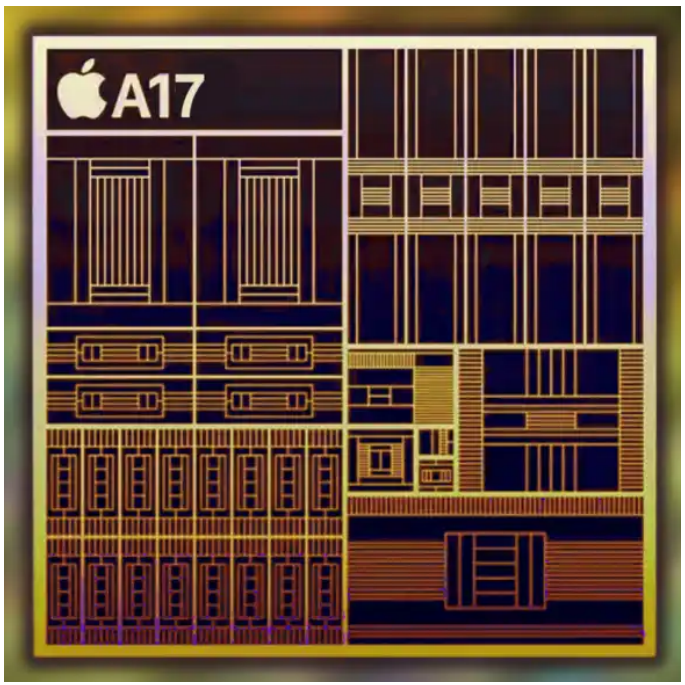
(See poll)

Trillion-dollar question

How do we make computers more powerful?

How do we make computers more powerful?

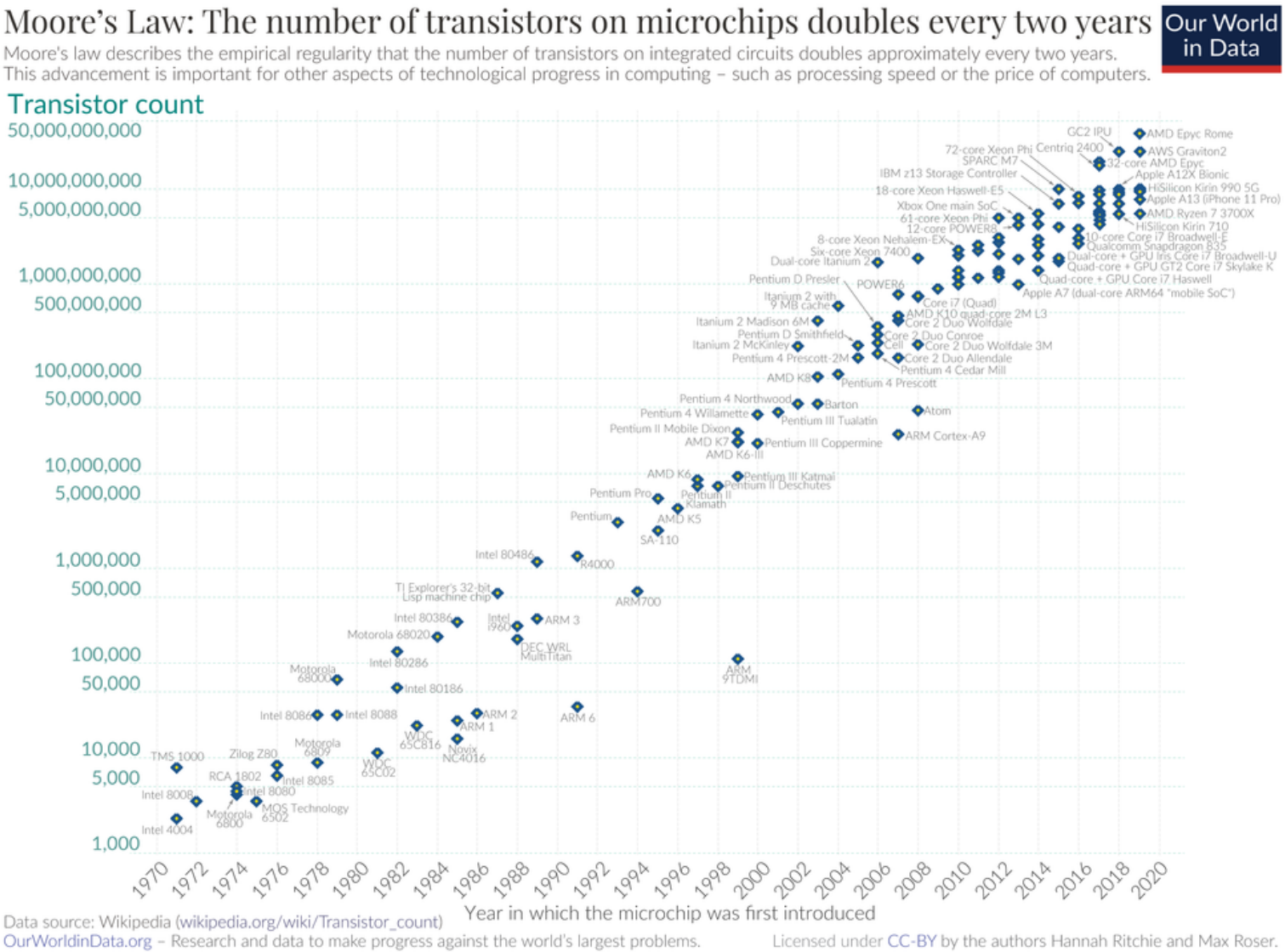
1. Make smaller transistors
2. Put more transistors onto a single chip



Apple A17: 19 billion transistors (iPhone 15 Pro / Pro Max, 2023)

Image: https://www.macworld.com/article/1532925/a17-bionic-preview-benchmarks-cpu-gpu-ram.html

How do we make computers more powerful?



How do we make computers more powerful?

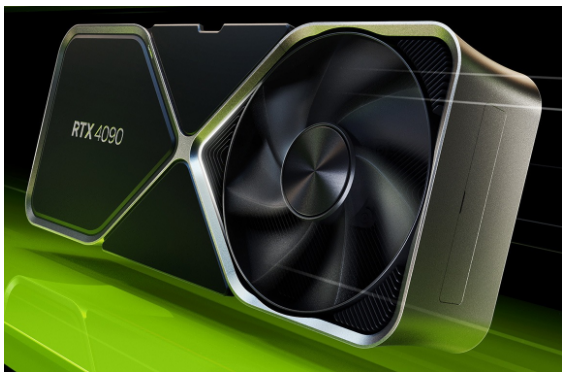
3. Use multiple processors in parallel

El Capitan (LLNL): ∼11 million cores (CPU+GPU), ∼1.742 exaFLOPS

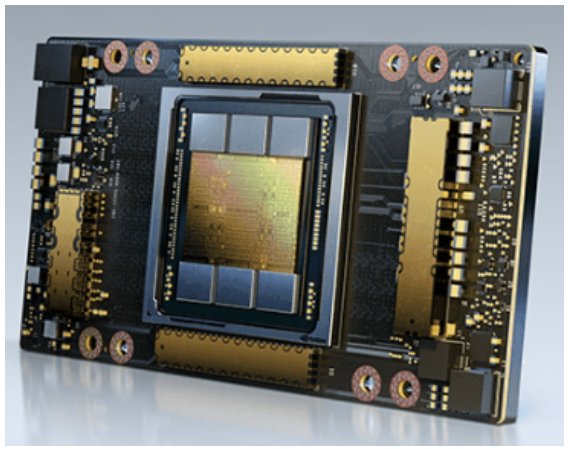
Image: https://asc.llnl.gov/exascale/el-capitan

How do we make computers more powerful?

4. Make special-purpose computers



GPU: NVIDIA RTX 4090



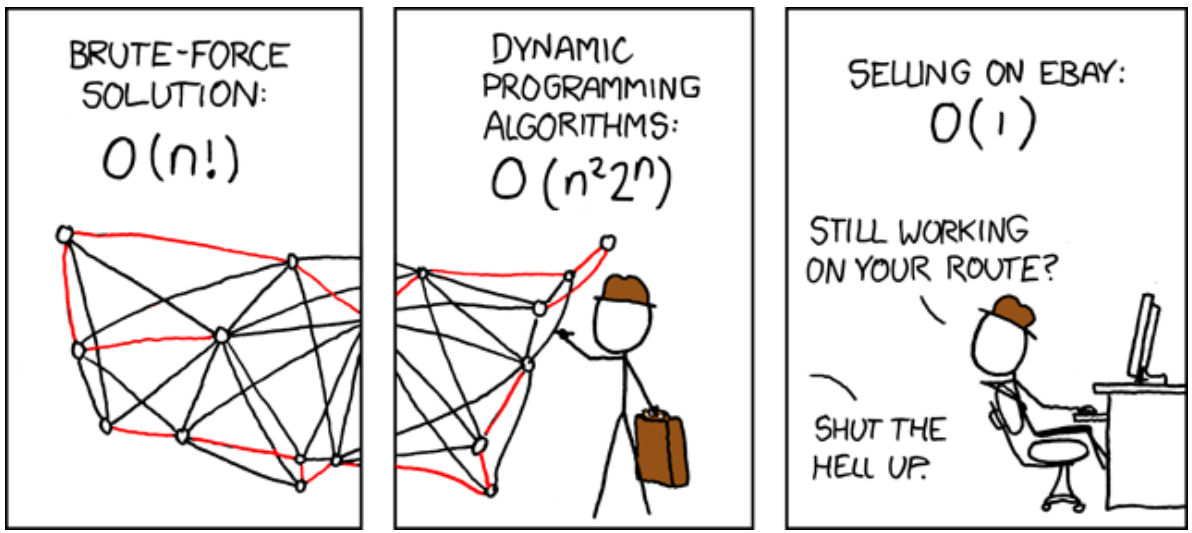
TPU: NVIDIA RTX A100

TPU: Google Trillium

Images:
https://www.nvidia.com/en-us/geforce/graphics-cards/40-series/rtx-4090/
https://www.nvidia.com/en-us/data-center/a100/
https://techcrunch.com/2024/05/14/googles-next-gen-tpus-promise-a-4-7x-performance-boost/

What's the point?

Some problems will remain intractable.



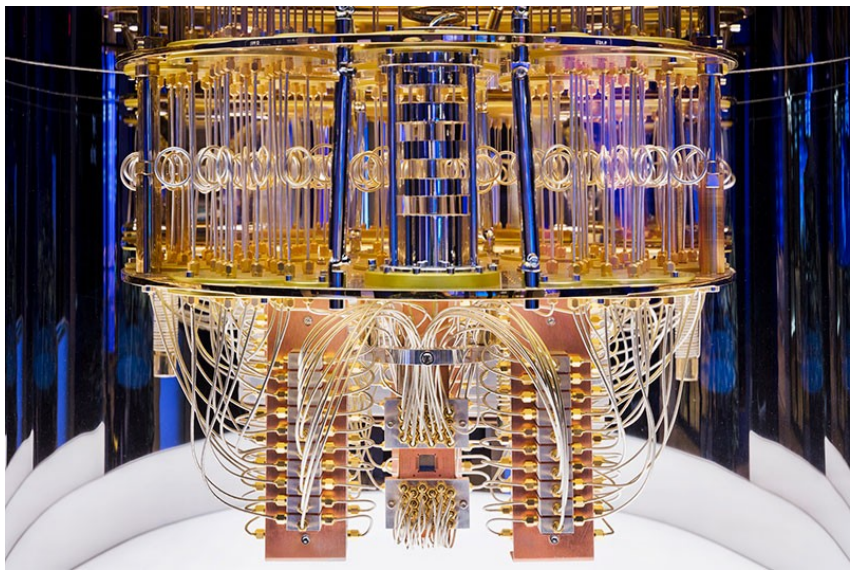
Sometimes that's good (example: our cryptographic infrastructure relies on assumptions about *computational hardness*).

But usually it just prevents us from doing interesting things.

Image: https://imgs.xkcd.com/comics/travelling_salesman_problem.png

How do we make computers more powerful?

4+. Make special-purpose computers *that work in a fundamentally different way*.



"Quantum computation and quantum information is the study of the information processing tasks that can be accomplished using quantum mechanical systems."

– Nielsen & Chuang

We know quantum algorithms with *exponential speedup* over the best-known classical algorithms (and some with lesser speedups).

Image: https://www.nature.com/articles/d41586-021-03476-5

Quantum computing

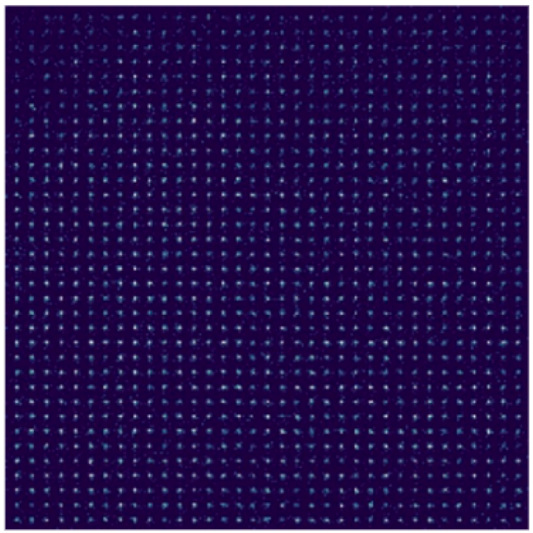
Lots of potential applications:

- Simulating and calculating properties of physical systems
- Searching large spaces
- Discrete mathematics, (breaking) cryptography
- Optimization
- Machine learning
- Things we haven't thought of yet!

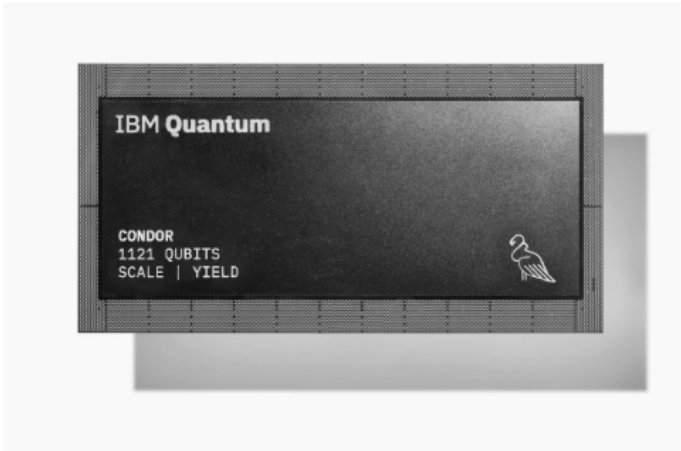
We can do proof-of-concept implementations of some algorithms...

Quantum computers

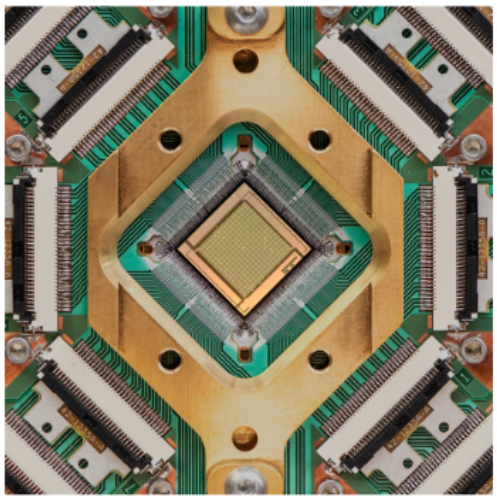
Largest machines (accurate as of 3 Sept)



1180 neutral atom qubits
Atom Computing



1121 superconducting qubits
IBM

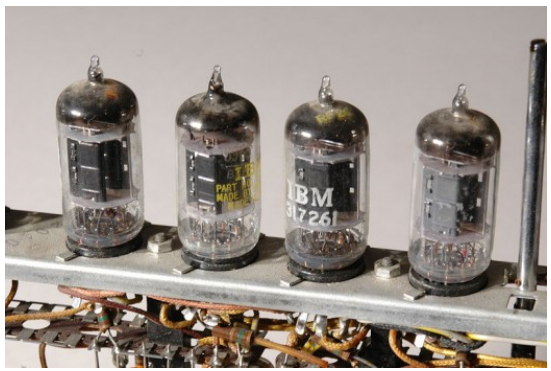


5000+ superconducting qubits
D-Wave

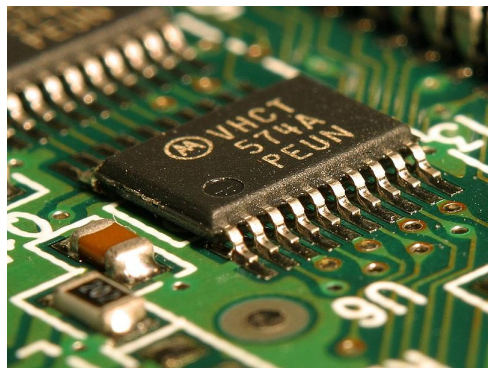
Images:
https://arstechnica.com/science/2023/10/atom-computing-is-the-first-to-announce-a-1000-qubit-quantum-computer/
https://techovedas.com/ibm-breaks-record-with-1121-qubit-quantum-processor-condor-takes-flight/
https://www.dwavesys.com/solutions-and-products/systems/

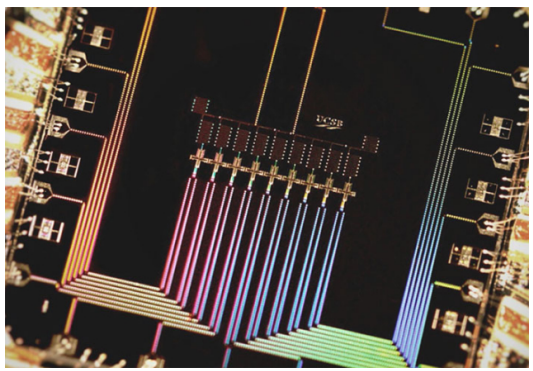
Quantum computing technologies

Too early to know which (combination?), if any, will win out.



vs.





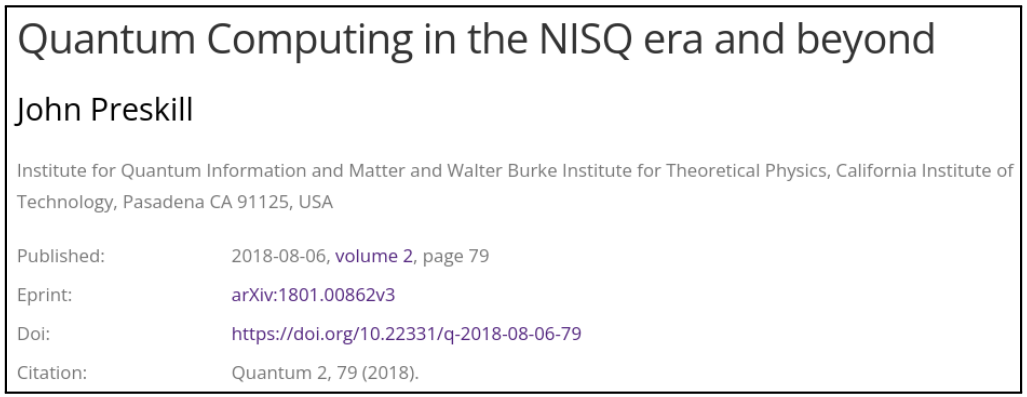
vs.

???

Images:
https://hubpages.com/business/What-Is-a-Transistor-and-Why-is-it-Important
https://en.wikipedia.org/wiki/Solid-state_electronics
https://physicsworld.com/a/google-gains-new-ground-on-universal-quantum-computer/

Quantum computers today

Termed **noisy, intermediate scale quantum** (NISQ) devices.



Quantum Computing in the NISQ era and beyond

John Preskill

Institute for Quantum Information and Matter and Walter Burke Institute for Theoretical Physics, California Institute of Technology, Pasadena CA 91125, USA

Published: 2018-08-06, volume 2, page 79
Eprint: arXiv:1801.00862v3
Doi: https://doi.org/10.22331/q-2018-08-06-79
Citation: Quantum 2, 79 (2018).

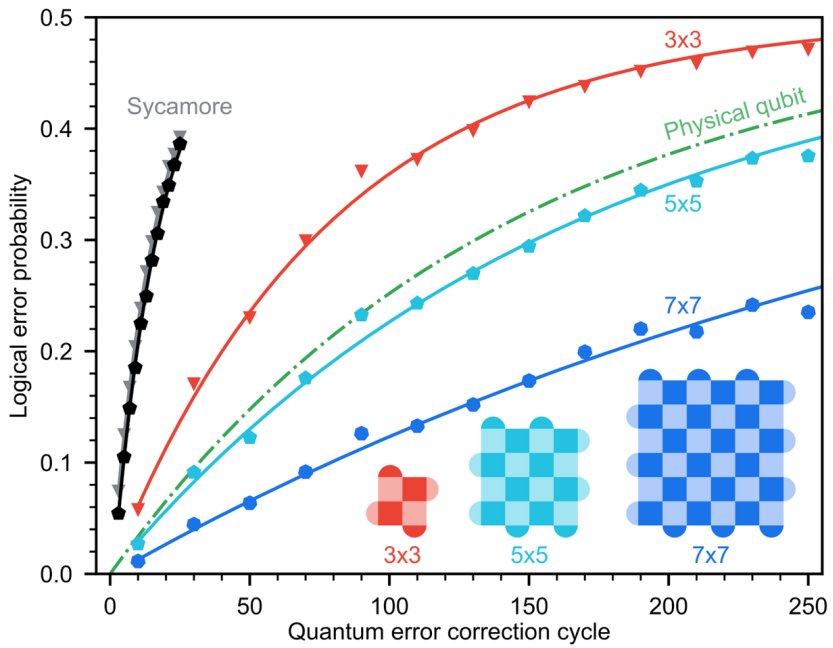
- Short coherence times
- High error rates (single-qubit, two-qubit operations)
- Limited qubit connectivity
- Challenging to scale up the hardware

But things are improving.

Quantum computing beyond the NISQ era

Solving problems "at scale" requires quantum error correction.

Have now seen multiple demonstrations of error-corrected qubits.
Example: Google's *Willow* (105 superconducting qubits).



By the last course module you'll know what everything in this plot is!

Image: https://research.google/blog/making-quantum-error-correction-work/

The path forward

IBM Quantum roadmap (late 2024)

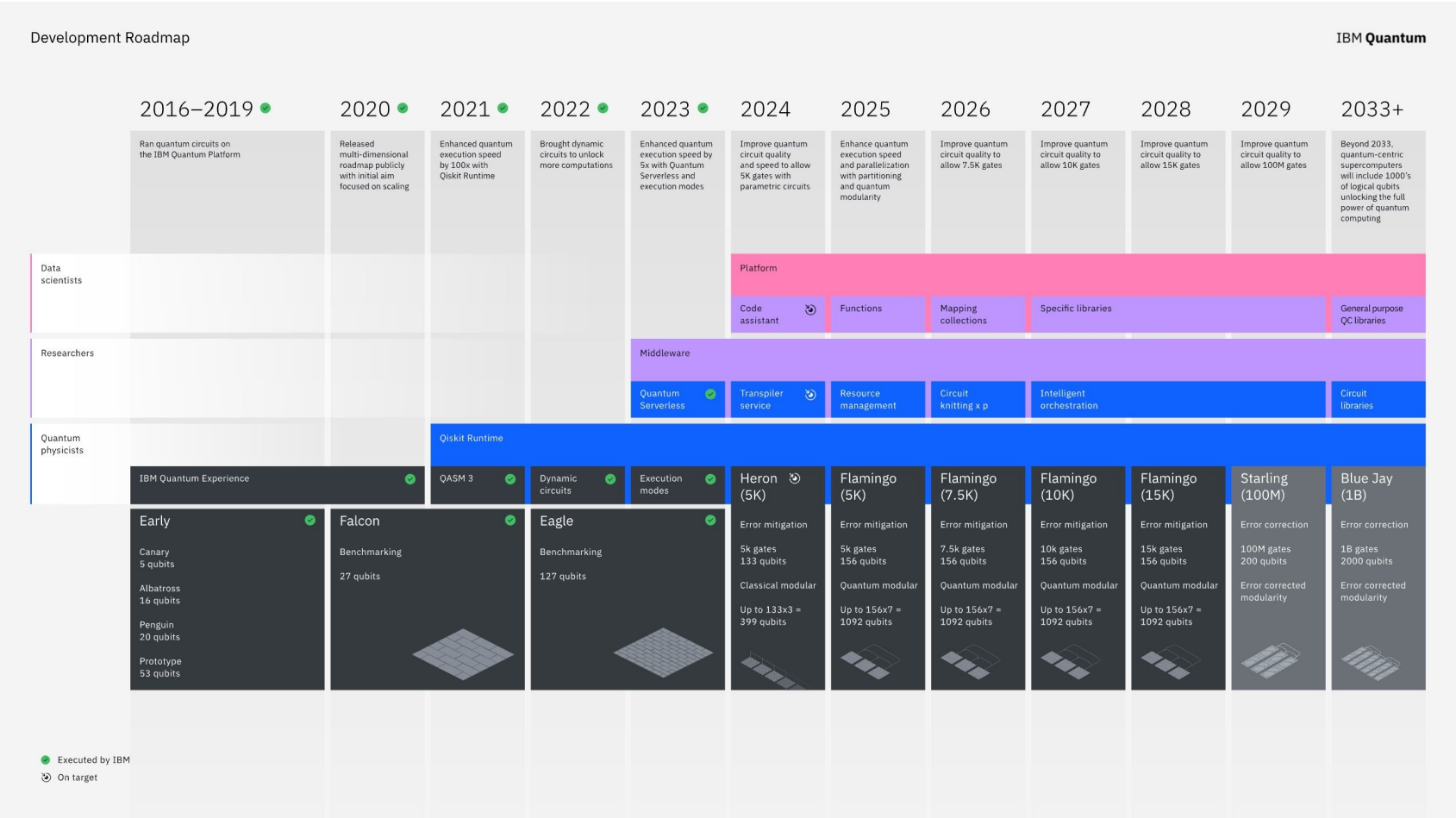


Image: https://www.ibm.com/quantum/blog/ibm-quantum-roadmap-2025

Course learning outcomes

Core goal: **learn how to program quantum computers** in a hands-on, software-focused setting.

- Describe the societal importance and implications of quantum computing
- Explain the theory and principles of gate-model quantum computing
- Outline and describe the operation of core quantum algorithms
- Implement textbook and research-level quantum algorithms using Python and PennyLane

In this course you will implement everything you learn!

Covered in this course

Divided into 5 modules:

1. Basic elements of quantum computation
2. Oracle-based algorithms, complexity, and quantum resources
3. Quantum Fourier transform-based algorithms
4. Quantum information theory
5. Quantum error correction

Differences from previous years:

- Improved tutorial contents
- More interactive lectures; emphasize conceptual understanding
- Technical assignments for practice (not credit)
- Written final exam

Logistics

My info:

Office:	KAIS 3043
Office hrs:	W 14:00-14:50, Fri 12:00-12:50, or by appointment. Open-door policy after 12:00 weekdays
Email:	`olivia@ece.ubc.ca`
GitHub:	`glassnotes`

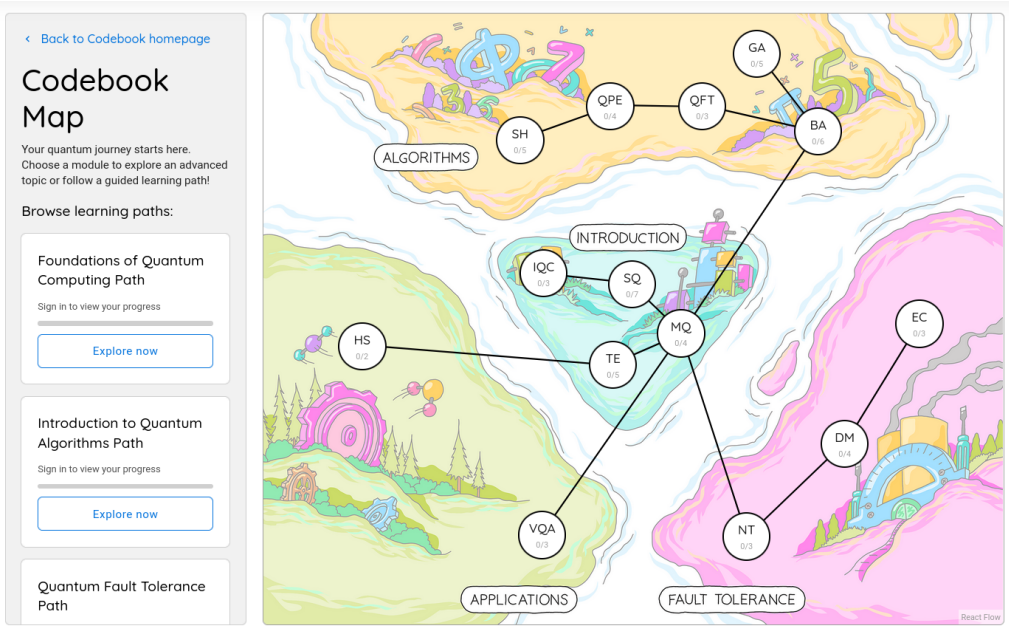
TA info: see syllabus and Piazza

All lecture slides/demos posted on course GitHub:

`https://github.com/glassnotes/CPEN-416`

Textbook

Primary resource: **free**, **online** PennyLane Codebook. Use to learn content, PennyLane, and do practice programming exercises.



https://pennylane.ai/codebook

Secondary (optional) resources listed in syllabus.

Assignments and grading

Five components:

- 10% tutorial assignments
- 10% in-class quizzes
- 30% in-class midterm exam
- 30% final group project
- 20% final exam

EECE 571O students: same grade breakdown, but some differences in project and exams (see syllabus).

Assignments and grading, cntd.

10% tutorial assignments (4):

- guided Jupyter notebooks with conceptual, technical, and coding questions
- done in small groups - start in tutorial, submit end of week
- what you submit must be your own work (and cite your collaborators!)

Purpose is to gain deeper understanding of content from class, and apply it to new situations.

No individual assignments for credit, but problems provided for practice. Assignment 0 available (math review, NumPy basics).

Assignments and grading, cntd.

10% quizzes (10):

- First one on Tuesday; in class, in PrairieLearn
- Individually, but open book/notes/docs
- Includes math and/or programming problems and conceptual questions; based on previous week's lecture material

Purpose is for reviewing content. Schedule is in syllabus.

Your lowest quiz grade will be automatically dropped.

Quiz questions will be made available for practice after class.

Assignments and grading, cntd.

30% midterm:

- Thursday 2 October during lecture
- pen-and-paper only; no programming
- closed-book and closed-notes (allowed one pre-approved, hand-written cheat sheet - details provided closer to date)

Purpose is to test knowledge of core content before studying advanced algorithms.

You must pass the midterm to pass the course

- can retry in form of oral exam (multiple attempts allowed)
- highest achievable score from retakes is 50%

Assignments and grading, cntd.

30% final project:

- Replicate results of a research paper
- Work in groups of 4
- Three components:
  - Fully-documented software implementation
  - Companion report
  - 15-20 minute in-class presentation
- Use a different quantum programming framework if you like

More details (rubric, grading, papers, etc.) after midterm.

Assignments and grading, cntd.

20% final exam:

- written during standard exam period
- technical and conceptual questions about basic course content
- questions about your group's project

You must write the exam to pass the course.

Academic misconduct and AI usage policies

The following will lead to a grade of 0 and/or a misconduct investigation:

- copying the work of classmates
- written content that closely paraphrases external sources
- usage of generative AI tools for any course component, including final project, unless indicated otherwise

Exceptions:

- if your IDE has, e.g., Copilot integration, you must acknowledge as source in assignment submission

Today

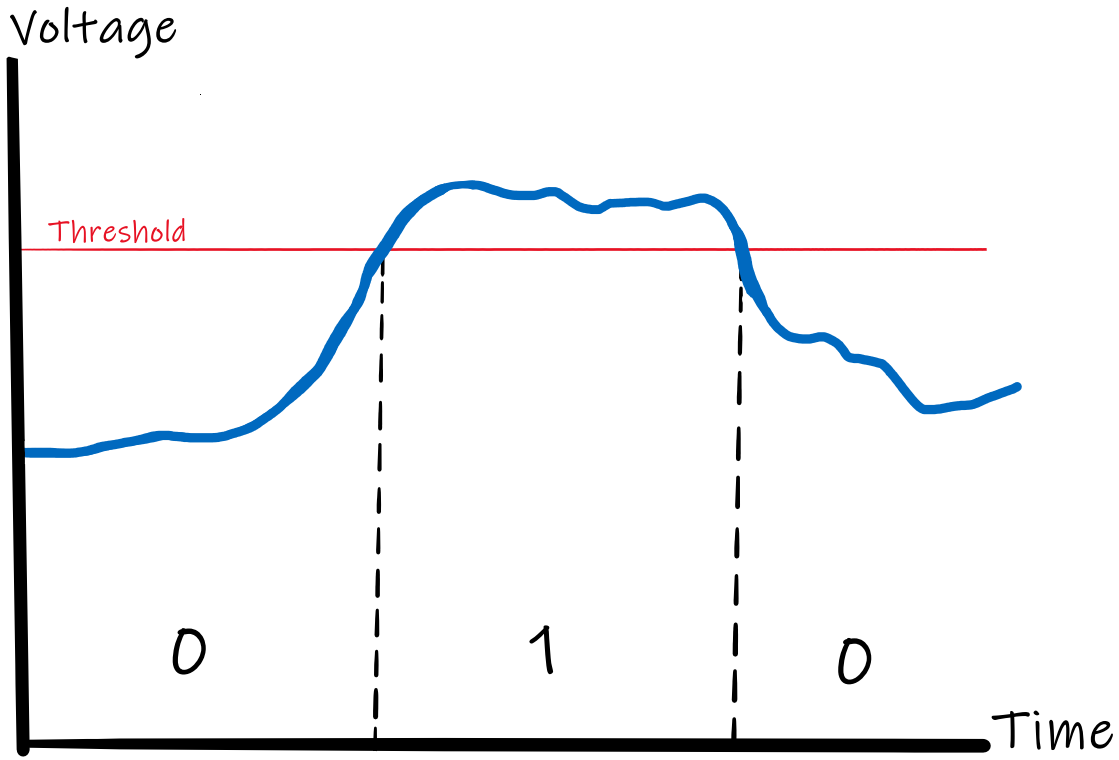
Learning outcomes:

1. Define quantum computing, and explain its societal importance
2. Define a *qubit* conceptually and mathematically
3. Outline the mathematical structure of the key components of quantum algorithms: states, gates, measurements

Bits

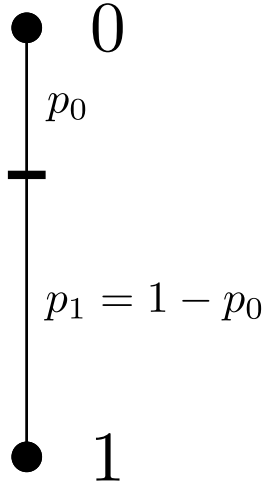
All computation in our computers today is done with bits.

The state of a bit is *either* 0 or 1.



Probabilistic bits

The value of a bit can be represented by a probability distribution.



Example: A coin flip: Heads (1) or Tails (0), each equally likely.

Example: Is it raining? Yes (1) or No (0) with some probability.

Probabilistic bits

Probability distributions can evolve over time.

Exercise:

- Suppose your laptop is connected to `ubcsecure`. The probability it will still be connected in an hour is 80%
- If it isn't connected now, the probability it will be connected in an hour is 60%

Your laptop is (presumably) currently connected. What is the probability it will still be connected in 3 hours?

Probabilistic bits

Represent the state of the system as a real, 2D **vector** ($\mathbb{R}^2$)

$$\text{Connection status} = \vec{s} = \begin{pmatrix} \text{Prob. connected} \\ \text{Prob. not connected} \end{pmatrix}$$

1. What properties does this state vector have?
2. If you're connected now, what is the state vector after 1 hour?
3. If you're not connected now, what is the state vector after 1 hour?
4. How does an arbitrary state vector transform after 1 hour?

Probabilistic bits

Represent the state of the system as a real, 2D **vector** ($\mathbb{R}^2$)

$$\text{Connection status} = \vec{s} = \begin{pmatrix} \text{Prob. connected} \\ \text{Prob. not connected} \end{pmatrix}$$

1. What properties does this state vector have?
   Values sum to 1
2. If you're connected now, what is the state vector after 1 hour?
   $\begin{pmatrix} 1 \\ 0 \end{pmatrix} \to \begin{pmatrix} 0.8 \\ 0.2 \end{pmatrix}$
3. If you're not connected now, what is the state vector after 1 hour?
   $\begin{pmatrix} 0 \\ 1 \end{pmatrix} \to \begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix}$
4. How does an arbitrary state vector transform after 1 hour?
   $\vec{s} \to \begin{pmatrix} 0.8 & 0.6 \\ 0.2 & 0.4 \end{pmatrix} \vec{s}$

Probabilistic bits

Now, determine what happens 1, 2, 3 hours from now...

$$\vec{s}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

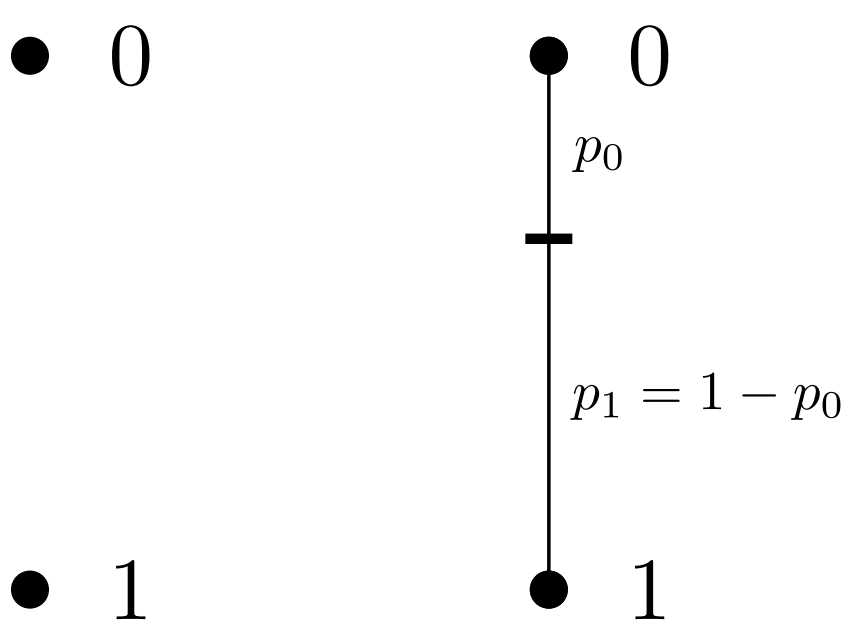
$$\vec{s}_1 = P\vec{s}_0 = \begin{pmatrix} 0.8 & 0.6 \\ 0.2 & 0.4 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.8 \\ 0.2 \end{pmatrix}$$

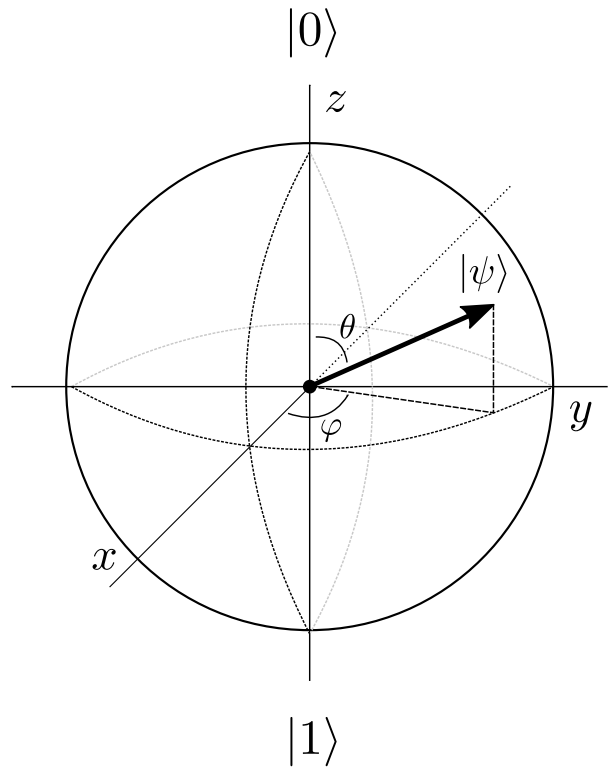
$$\vec{s}_2 = P\vec{s}_1 = \begin{pmatrix} 0.8 & 0.6 \\ 0.2 & 0.4 \end{pmatrix} \begin{pmatrix} 0.8 \\ 0.2 \end{pmatrix} = \begin{pmatrix} 0.76 \\ 0.24 \end{pmatrix}$$

$$\vec{s}_3 = P\vec{s}_2 = \begin{pmatrix} 0.8 & 0.6 \\ 0.2 & 0.4 \end{pmatrix} \begin{pmatrix} 0.76 \\ 0.24 \end{pmatrix} = \begin{pmatrix} 0.752 \\ 0.248 \end{pmatrix}$$

Qubits

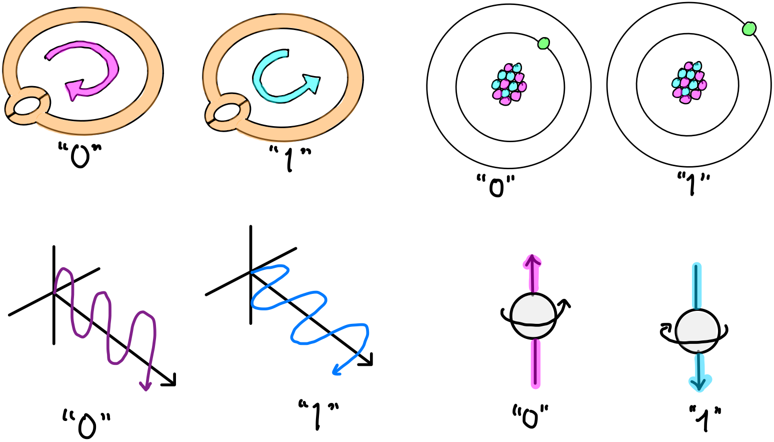
Extension of probabilistic bits to 2-level quantum systems:





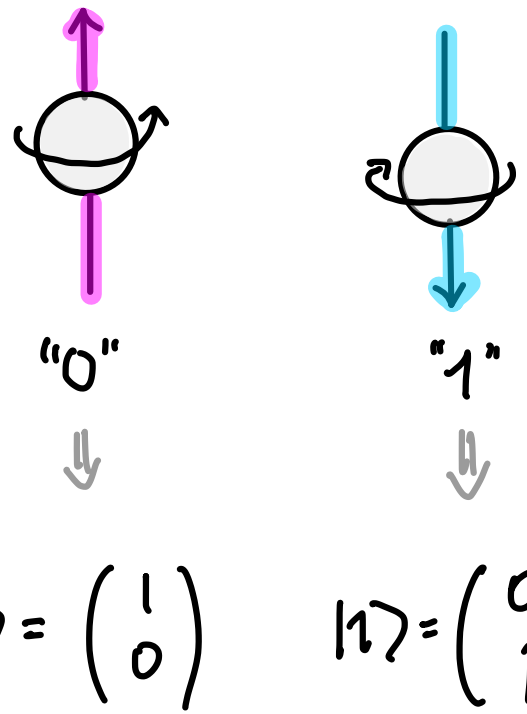
Qubits

Quantum computers work by manipulating **qubits**.



Mathematical representation of qubits

The vector space where qubits live is called **Hilbert space**.

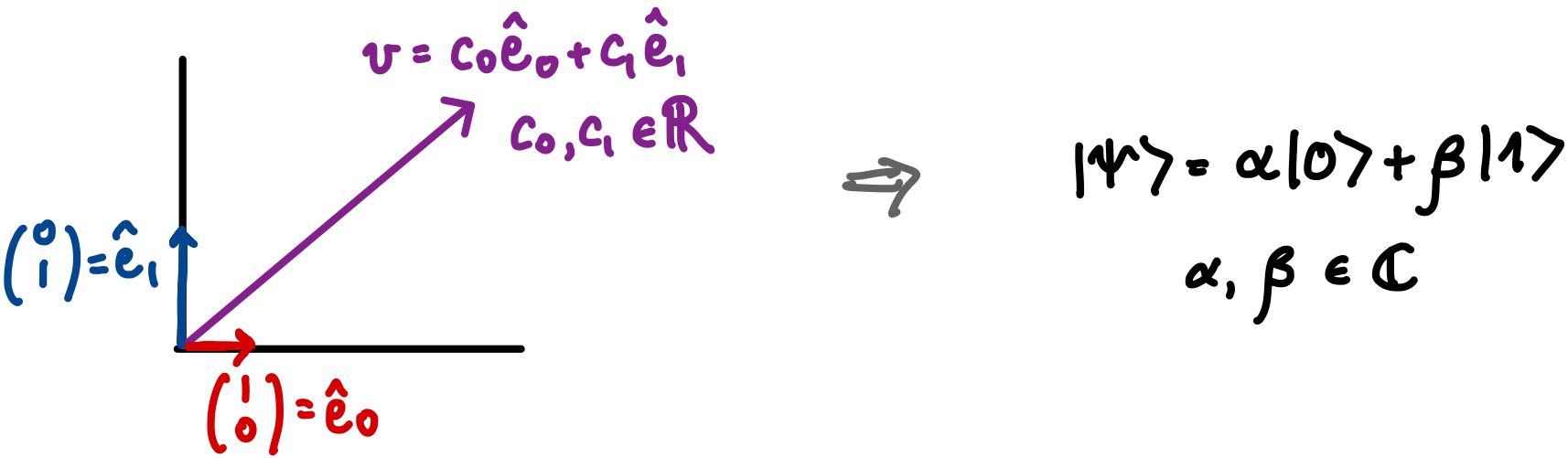


These two vectors together form the **computational basis**.

The notation $|\cdot\rangle$ is called Dirac, or **bra-ket notation**. $|\cdot\rangle$ is a ket.

Mathematical representation of qubits

Qubit states are linear combinations of these basis states.



The coefficients α and β can be complex, and are called **amplitudes**. "Valid" qubit state vectors have unit length:

$$|\alpha|^2 + |\beta|^2 = \alpha\alpha^* + \beta\beta^* = 1, \quad \alpha, \beta \in \mathbb{C}.$$

Such states are called **normalized**.

Mathematical representation of qubits

Exercise: is $|\psi\rangle = \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}e^{0.2i}|1\rangle$ a valid quantum state?

$$\alpha = \frac{1}{\sqrt{3}} \rightarrow \qquad |\alpha|^2 = \alpha\alpha^* = \frac{1}{3}$$

$$\beta = \sqrt{\frac{2}{3}}e^{0.2i} \rightarrow \qquad |\beta|^2 = \beta\beta^* = \sqrt{\frac{2}{3}}e^{0.2i}\sqrt{\frac{2}{3}}e^{-0.2i} = \frac{2}{3}$$

Mathematical representation of qubits

A qubit in a linear combination

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

is in a **superposition** of the two basis states.

Note: a qubit in superposition isn't *"in both states simultaneously"*!

Using qubits for computation

1. **Prepare** qubits in a **superposition**
2. Apply **operations** that **entangle** the qubits and manipulate the amplitudes
3. **Measure** qubits to extract an answer

...how do we do this?

Manipulating qubits: unitary operations

Single-qubit states are manipulated by 2×2 unitary matrices. A matrix U is unitary if

$$U^\dagger U = UU^\dagger = \mathbb{1}.$$

The $\dagger$ means "conjugate transpose":

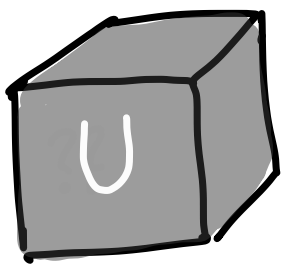
$$U^\dagger = (U^*)^T.$$

Unitary operations, or **gates**, are **reversible**: if we apply U, we can undo it by applying $U^\dagger$.

Manipulating qubits: unitary operations

Unitary operations preserve the length of vectors, i.e., *maintain the normalization of qubit states*.

$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ $\quad |\alpha|^2 + |\beta|^2 = 1$



$UU^\dagger = U^\dagger U = \mathbb{1}$

$|\psi'\rangle = \alpha'|0\rangle + \beta'|1\rangle$ $\quad |\alpha'|^2 + |\beta'|^2 = 1$

Will prove on your first assignment

Manipulating qubits: unitary operations

Unitary operations act *linearly* on superpositions:

$$U(\alpha|0\rangle + \beta|1\rangle) = \alpha U|0\rangle + \beta U|1\rangle$$

This is a key contributor to the power of quantum computing!

Exercise: The matrix

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

is unitary. Determine its action on the basis states.

$$\begin{aligned} X|0\rangle &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle, \\ X|1\rangle &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle. \end{aligned}$$

Example: X

Exercise: What does X do to the state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

$$\begin{aligned} X\left(\alpha|0\rangle + \beta|1\rangle\right) &= \alpha \cdot X|0\rangle + \beta \cdot X|1\rangle \\ &= \alpha|1\rangle + \beta|0\rangle. \end{aligned}$$

X is also known as Pauli X, the bit flip, or NOT gate. It is *self-inverse*.

Example: H

Perhaps the most important unitary in quantum computing is the Hadamard gate (H):

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

This gate creates a *uniform superposition*:

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) = |+\rangle \\ H|1\rangle &= \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) = |-\rangle \end{aligned}$$

Example: H

What if we apply it twice?

$$\begin{aligned} H|+\rangle &= H \cdot \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ &= \frac{1}{\sqrt{2}}(H|0\rangle + H|1\rangle) \\ &= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right) \\ &= |0\rangle \end{aligned}$$

Example: Z

The gate

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

does something a little different.

Apply to basis states:

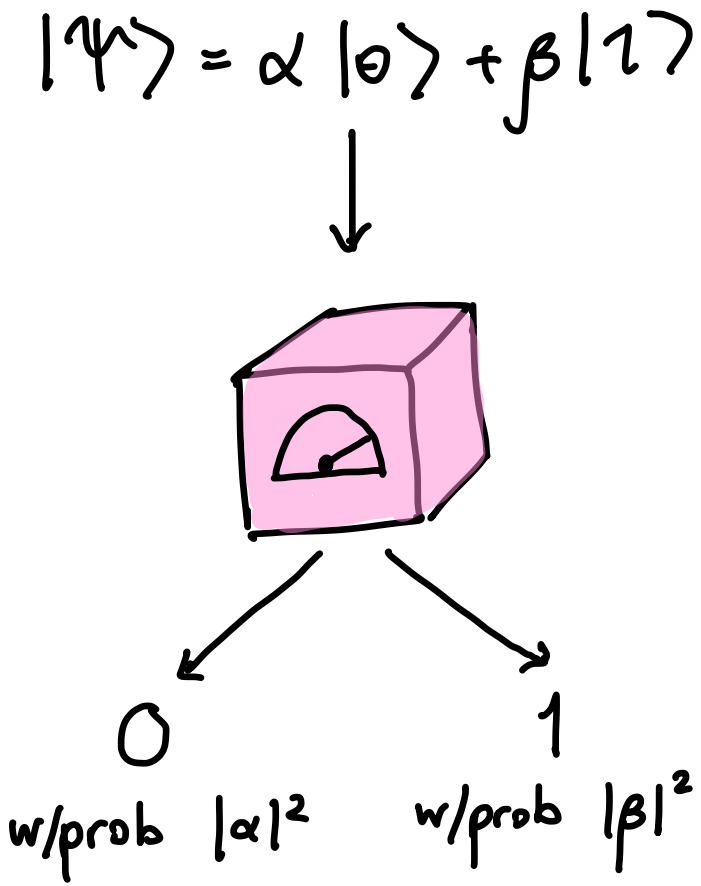
$$\begin{aligned} Z|0\rangle &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle, \\ Z|1\rangle &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -\begin{pmatrix} 0 \\ 1 \end{pmatrix} = -|1\rangle. \end{aligned}$$

Just changes the sign. (This will be important later!)

Measuring qubits

Manipulating qubits changes the amplitudes. Amplitudes determine probability of observing the qubit in $|0\rangle$ or $|1\rangle$ when we measure (it's **probabilistic**!).

Measuring gives us a single bit of information (0 or 1). To get more, we need to run the algorithm multiple times (multiple **shots**).



Single-qubit gates: applying sequences of gates

Exercise: A qubit starts in state $|0\rangle$. What is the probability of observing the qubit in state $|1\rangle$ after applying H, then Z?

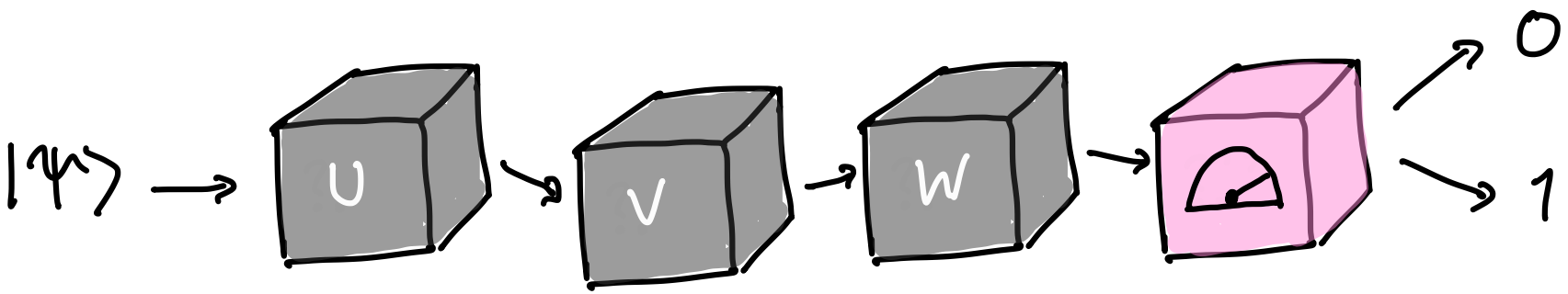
$$\begin{aligned} ZH|0\rangle &= Z\left(\frac{|0\rangle}{\sqrt{2}} + \frac{|1\rangle}{\sqrt{2}}\right) \\ &= \left(\frac{|0\rangle}{\sqrt{2}} - \frac{|1\rangle}{\sqrt{2}}\right) \end{aligned}$$

$$\text{Prob}(|1\rangle) = \left(\frac{1}{\sqrt{2}}\right)^2 = 0.5$$

Quantum computing

Quantum computing is the act of manipulating the state of qubits in a way that represents solution of a computational problem:

1. **Prepare** qubits in a **superposition**
2. Apply **operations** that **entangle** the qubits and manipulate the amplitudes
3. **Measure** qubits to extract an answer



For next time

Monday's tutorial: review of math concepts (linear algebra, complex numbers), and practice today's concepts

Content:

- Getting started with PennyLane
- Quantum circuits
- More unitary operations

Action items:

1. Set up Python environment with PennyLane v0.42.3
2. See instructions on Canvas to access PrairieLearn and Piazza
3. Do assignment 0 for practice / review

Recommended reading:

- Codebook modules IQC and SQ.
- Nielsen and Chuang 1.1-1.2