Quantum computation and compilation

Matthew Amy

Simon Fraser University

CS Undergraduate Research Symposium
April 11, 2022

Computation

Computation is a **physical** process

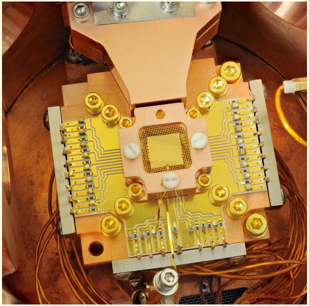
We use **abstractions** to describe and model computation

- 0 for low voltage, 1 for high voltage
- Turing machines

The (extended) Church–Turing thesis:

> *A probabilistic Turing machine can efficiently simulate any* physical *model of computation.*

Quantum computation



Classical models of computation are based on **classical** physics.

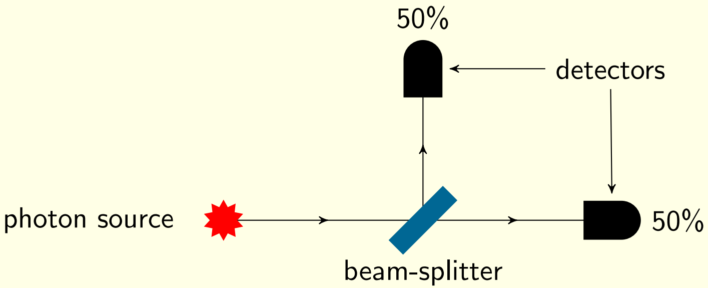
Richard Feynman, 1982:

> *A classical computer cannot efficiently simulate a quantum mechanical system.*

Subsequent algorithms using **quantum effects** for speed-ups:

- (Shor, 1994) Integer factorization
- (Lloyd, 1996) Simulation of quantum systems
- (Grover, 1996) Unstructured search
- Discrete logarithms, linear systems, knot invariants, . . .

Beam-splitters

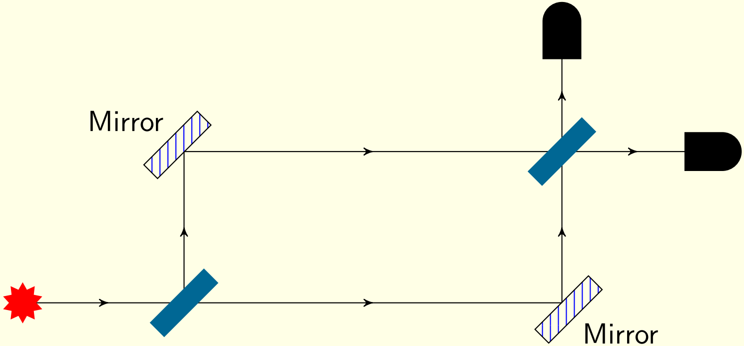


A beam-splitter acts as a classical **coin flip**: a photon traveling through it will either

- continue straight through, or
- be reflected

with **equal probability**.

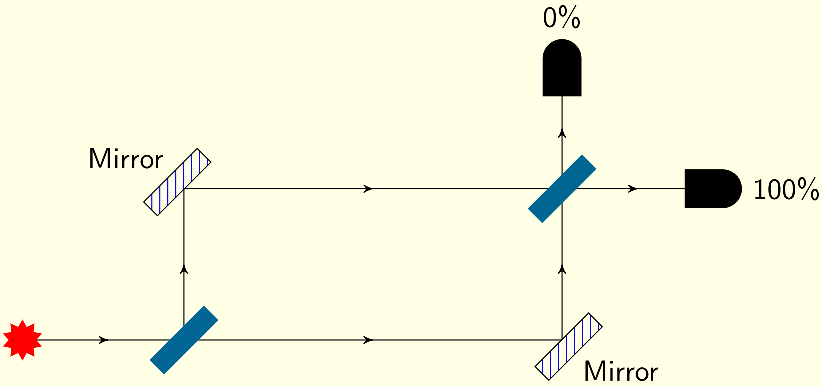
The beam-splitter experiment



Where will a single photon be detected?

- Classical intuition says equal probability at either location

The beam-splitter experiment

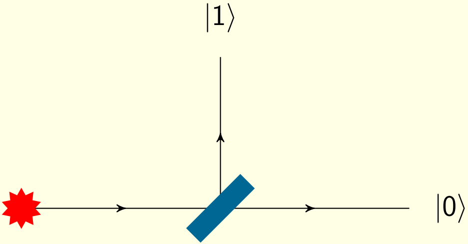


Experimentally it will always appear at the lower detector

- Intuition is that the photon took **both paths simultaneously**
- **Interference** causes paths to the upper detector to cancel

How do we model this abstractly?

The linear algebraic model



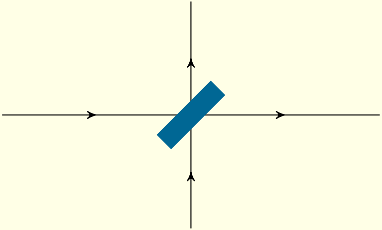
We map the **classical states** to a basis of $\mathbb{C}^2$

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

The state of a qubit is a unit vector $|\psi\rangle \in \mathbb{C}^2$, corresponding to a **superposition** of the classical 0 and 1 states

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

Quantum gates



Transformations on a quantum state are **unitary** operators $U : \mathbb{C}^2 \to \mathbb{C}^2$ called **gates**

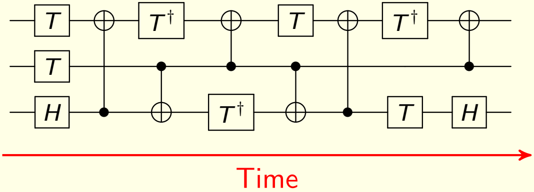
$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Gates transform states via matrix multiplication

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

Quantum circuits

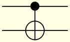
Large unitaries are built by composing gates in **circuits**



Common gates:

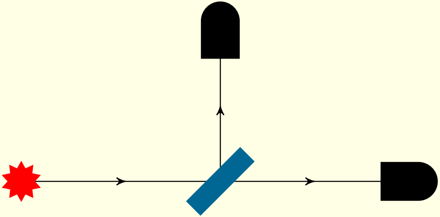
$$S = -\boxed{S}- = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \quad H = -\boxed{H}- = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

CNOT =



$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad T = -\boxed{T}- = \begin{bmatrix} 1 & 0 \\ 0 & \omega := e^{i\frac{\pi}{4}} \end{bmatrix}$$

Measurement



When we **measure** a qubit in a superposition

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

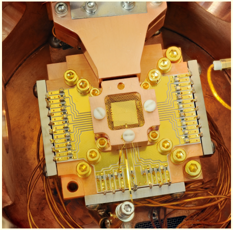
the state collapses to

- $|0\rangle$ with probability $|\alpha|^2$
- $|1\rangle$ with probability $|\beta|^2$

The measurement probabilities form a **probability distribution**, forcing $|\psi\rangle$ to be a unit vector:

$$|\alpha|^2 + |\beta|^2 = 1$$

Quantum programs



- States: $x \in \{0,1\}^n = \mathbb{Z}_2^n$
- Functions: $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^m$

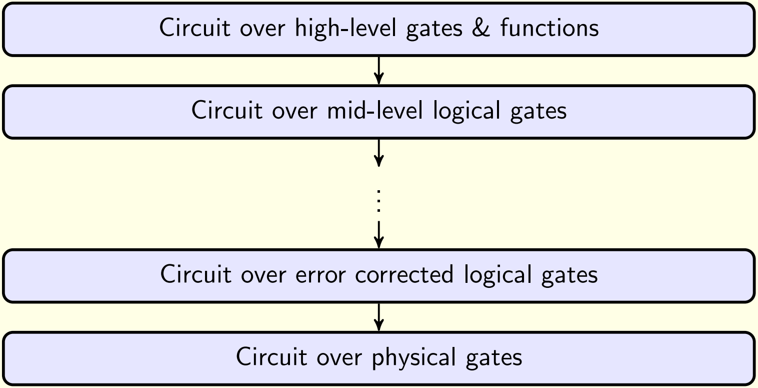
- States: $|\psi\rangle \in \mathbb{C}^{2^n}$
- Functions: $U : \mathbb{C}^{2^n} \to \mathbb{C}^{2^n}$

Typical quantum program:

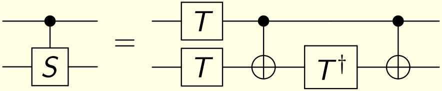
1. Apply some circuit to the $|00\ldots0\rangle$ state
2. Measure some or all of the qubits
3. Process the results on a classical computer

How do we program & compile circuits?

Quantum compilation



Compile by replacing gates with lower-level circuits, e.g.,



Gate sets

A set of gates is **universal** if it can **approximate** any unitary up to arbitrary accuracy.

Theorem (The Solovay-Kitaev theorem)

Given a set $\mathcal{G}$ of gates which is dense in $SU(2)$, any $U \in SU(2)$ can be approximated to within ϵ error using a poly-logarithmic number of gates taken from $\mathcal{G}$.

$\{H, CNOT, T\}$ **is the standard error corrected universal set**

Algebraic compilation

Algebraic compilation = compile using algebraic characterizations.

The **number-theoretic method**:
Let $\mathbb{D} = \{a/2^k | a, k \in \mathbb{Z}\}$ be the ring of Dyadic fractions and let

$$\mathbb{D}[\omega] = \{a\omega^3 + b\omega^2 + c\omega + d \mid a, b, c, d \in \mathbb{D}\}$$

where $\omega = e^{\pi i/4}$.

(Kliuchnikov et al. 2013, Giles & Selinger 2013):

A $2^n \times 2^n$ unitary matrix U can be written as a product of $\{H, CNOT, T\}$ gates if and only if U has entries in $\mathbb{D}[\omega]$.

(Amy, Glaudell, & Ross 2020):

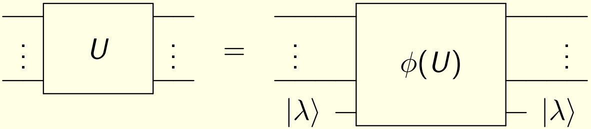
Similar characterizations for $\mathbb{D}, \mathbb{D}[\sqrt{2}], \mathbb{D}[i\sqrt{2}]$, and $\mathbb{D}[i]$

Number-theoretic embeddings

Let $\mathcal{R}$ be a ring and $\mathcal{R}[\alpha]$ be an **algebraic** extension of $\mathcal{R}$.

(Amy, Glaudell, Ross, et al. 2022):

> *If there exists a pseudo-companion matrix $\Gamma \in \mathcal{M}_{k\times k}(\mathcal{R})$ for α, then any $n \times n$ unitary $U \in \mathcal{M}_{k\times k}(\mathcal{R}[\alpha])$ can be embedded over $\mathcal{R}$ with a suitable* **resource state**.



The phase polynomial method

Circuits over **restricted** or **non-universal** gate sets are often easier to efficiently characterize & compile.

(Amy, Maslov, & Mosca, 2014)

Any n-qubit circuit over $\{CNOT, X, T\}$ *can be written as*

$$U|\mathrm{x}\rangle = \omega^{P(\mathrm{x})}|A\mathrm{x}\rangle$$

where $A \in \mathcal{M}_{n\times n}(\mathbb{Z}_2)$ *and* P *is a* **phase polynomial***:*

$$P(\mathrm{x}) = \sum_{\mathrm{y}\in\mathbb{Z}_2^n} a_{\mathrm{y}}(x_1y_1 \oplus x_1y_2 \oplus \cdots \oplus x_ny_n)$$

Phase polynomial synthesis problems

Given a phase polynomial

$$P(\mathrm{x}) = \sum_{\mathrm{y}\in\mathbb{Z}_2^n} a_{\mathrm{y}}(x_1y_1 \oplus x_1y_2 \oplus \cdots \oplus x_ny_n)$$

can we synthesize with the minimal...

(**Amy, Maslov, & Mosca 2014**) T-depth

Poly-time via reduction to Matroid partitioning.

(**Amy, Azimzadeh, & Mosca 2018**) CNOT gates

Unique combinatorial problem, NP-hard in some cases.

(**Amy & Mosca 2019**) T gates

Poly-time equivalent to decoding $\mathcal{RM}(n-4, n)^*$.

Just the tip of the iceberg...

- Near term/non-fault-tolerant computers
- Compilation & error correction co-design
- **Symbolic synthesis**
- ZX-calculus compilation
- **Cost lower bounds**
- Optimal reversible circuit synthesis
- **Relative-phase implementations**
- **Measurement-assisted circuits**
- Pebbling strategies
- **Applications of number-theoretic embeddings**
- Algorithm-specific compilation problems
- Pauli-based computing
- ???

Thank you!

I'm looking for students!