Lecture 1

Course webpage: `https://wdaochen.com/teaching_undergrad_2025w1`.

Bits and qubits Classical computers use bits that operate according to the rules of classical mechanics, quantum computers use "quantum bit" (qubits) that operate according to the rules of quantum mechanics.

Bits are usually instantiated as current being flowing or not flowing in a transistor. Qubits can be instantiated in many ways, e.g.,

1. Atoms: ground and excited states of electrons in the atom. (QuEra).
2. (Dual-rail) photons: left and right (rail) spatial locations of the photon (PsiQuantum).
3. Superconducting circuits (artificial atoms): ground and excited states of the current (always flowing) in the circuit (Google, IBM).

The point to stress is:

Quantum computers are not "sped-up versions" of classical computers, e.g., by "making the current run faster", but a new type of computer that uses different (more general) laws of physics.

What makes a good qubit? A good qubit needs to satisfy two requirements in tension with each other:

1. The qubit does not want to interact with the environment.
2. The qubit does want to interact with the experimenter wishing to perform a computation.

These are in tension because in practice it is hard for the experimenter to distinguish itself from the environment. Atoms, photons and other instantiations of qubits in nature score highly on the first point and less highly on the second point. For superconducting circuits the reverse is true.

States of bits and qubits. n bits can be in one of 2^n states. In contrast n qubits can be in one of those states but also states "in-between" that are called superposition states. As a brief preview: the state of n qubits is mathematically written as

$$\sum_{x\in\{0,1\}^n} \alpha_x |x\rangle , \tag{1}$$

where the α_xs are complex numbers. This can also be written as a *vector* $(\alpha_{0\ldots0}, \ldots, \alpha_{1\ldots1})$ if you like.

But it is absolutely critical to understand the following: by employing random coin flips, bits can also be in "classical superposition states" that can be written as

$$\sum_{x\in\{0,1\}^n} p_x \boxed{x}, \tag{2}$$

where the p_x are non-negative real numbers. For example, the state of n fair coin flips can be described as

$$\sum_{x\in\{0,1\}^n} \frac{1}{2^n} \boxed{x}, \tag{3}$$

The fact that there is a "superposition" is not what accounts for the difference between classical and quantum computation.

The main difference is that the α_x can be complex[1] whereas the p_x are non-negative.

Put another way: the state vector is of length 2^n in deterministic, randomized, quantum computation – this is not the distinguishing feature of quantum computation. The distinguishing feature is that the state vector can contain complex numbers in the quantum case.

[1] In fact, there is a sense in which the only difference is that the α_x can be *negative*: complex numbers can be viewed as simply a good accounting method to keep track of all the minus signs. See the intro to my thesis for more details.

Example. Consider $n = 3$ bits.

1. Deterministic computation: a bitstring 010 can be represented as the state vector $[0, 0, 1, 0, 0, 0, 0, 0]$ if we agree on the following indexing:

$$\begin{array}{cccccccc} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ \hline 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array} \tag{4}$$

2. Randomized computation = deterministic computation + ability to flip fair coins. The state vector of three fair coins before looking is $[1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8, 1/8]$ (under the same indexing as above, will stop saying this).

Exercise that we did in class: suppose I XORed the second bit onto the third bit, i.e., $abc \mapsto ab(\mathrm{XOR}(b, c))$, what happens to the state vector? In the randomized case, we considered the initial state corresponding to the first two bits being fair coin flips and the last one being fixed to 0. Note

$$\mathrm{XOR}(a, b) \coloneqq a + b \mod 2 \quad \text{for all } a, b \in \{0, 1\}. \tag{5}$$

(I use $\coloneqq$ to mean "is defined to be" or "by definition is".)

Double-slit experiment

Or: what was the origin of the complex (negative) numbers. Without this experiment, it is doubtful that people would have come up with the idea of having state vectors involving negative numbers on their own.

1. Do not conflate Young's double-slit experiment with the later quantum double-slit experiment. The former (1801) only tracks intensity of light and can be simply explained by modelling light as a wave. The latter (1927) tracks *individual* particles – initially electrons but very recently even with *individual* photons apparently: MIT news (July 28 2025). We always refer to the quantum version.

2. Ignoring the part of "observing which slit the election went through changes the outcome", the experiment is already counterintuitive: when only one slit is open, there exists a position x^* (in fact, multiple such) where the electron can land at that are not possible when two slits are open.

3. The above fact is a first hint at the concept of *subtraction*, aka, negativity. The modern viewpoint: the state of the electron emitted from the first slit *subtracted* the state of the electron emitted from the second slit to give 0 as the overall state at x^*. Comment: I might put something more about this on HW1. If you want more details now, I highly recommend The Feynman Lectures on Physics, Volume III, Chapter I: `https://www.feynmanlectures.caltech.edu/III_01.html`.