CS M238 Lecture Notes

Kyle Chui

2022-09-22

Contents

1 Lecture 1

1.1 History

Quantum computers only really came into existence in 2016, and modern quantum computers have 127 qubits. We hope that the number of qubits explodes in the next few years, with Google targeting 1 million qubits by 2029.

We will *never* be able to simulate 100+ qubit quantum computers, since the state space just gets too large. Modern supercomputers can simulate a maximum of 70 qubits (Summit).

Normal computers typically have an error rate of once a year, which is mitigated by checksums and other mechanisms. To contrast, the error rate for a quantum computer is around 1%, which is a lot higher. Furthermore, we need around 1000 qubits in order to "validate" that a single qubit is correct. We call these validated qubits "perfect qubits".

Definition. *Neven's Law*
Quantum computers are gaining computational power at a doubly exponential rate.

Quantum computers are *very good* at solving linear algebra problems, e.g. machine learning or physics simulations, not *all* problems. According to experts, roughly 6000 perfect qubits are needed to be better than a classical computer.

We can't perform that many computations per qubit because of their decoherence time.

1.2 Basic Mathematics

A qubit can be represented as a unit vector in $\mathbb{C}^2$. Every computation step takes a unit vector to a unit vector, i.e. they can be represented by unitary matrices. Quantum computing uses complex numbers to allow amplitudes (read: probabilities) to be "negative", while still having positive norm.

To convert amplitudes to probabilities, we use Born's law and take the square of the norm, i.e. the probability of an event with amplitude $-\frac{1}{\sqrt{2}}$ is

$$\left(-\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2}.$$

1.3 Abstractions

Classical computing languages mostly differ in how they deal with abstractions; different choices lead to different languages. At the moment, quantum computing has little to no abstractions, and the languages are all more similar (this course uses Qiskit).

While in classical computing we can choose any two registers to perform some multi-register computation, for quantum computers there are more restrictions. Operations can only be performed on neighboring qubits, as opposed to any two arbitrary registers.

2 Lecture 2

	Classical Computing	Quantum Computing
Software	Boolean Algebra	Linear Algebra
Hardware	Classical Mechanics e.g. semiconductors	Quantum Mechanics e.g. superconductors

2.1 The Four Postulates of Quantum Mechanics

2.1.1 State Space Rule

The state space is a unit vector with dimension 2^n, where n is the number of qubits. We use the following abbreviations:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{and} \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

The state space rule says that we are working with superpositions (or linear combinations).

2.1.2 Composition Rule

We use the tensor product to compose objects; for each additional qubit the *dimension* of our state space doubles.

2.1.3 Step Rule

We use unitary matrices to get from one state to the next state. This is because unitary matrices send unit vectors to unit vectors. For example,

$$U|\psi\rangle = |\varphi\rangle.$$

Note that this means that the size of U is $2^n \times 2^n$.

2.1.4 Measurement Rule

There is a way to get information out of a quantum computer. When we measure a computation performed with n qubits, we get n bits out of the system.

2.2 Computations

For probabilistic programming, our state is no longer a bit vector, but rather a probability vector. The matrix used to update this state is called a stochastic matrix. We can take the tensor product of two vectors to get the next state.

$$\begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix}.$$

	Probabilistic Computing	Quantum Computing
Numbers	Real	Complex
State	Vector of Probabilities $\sum p_i = 1$	Vector of Amplitudes $\sum \lvert a_i \rvert^2 = 1$
Step	Stochastic Matrix	Unitary Matrix

The Hadamard matrix is given by

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

Observe that

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

We also have

$$\begin{aligned} H|1\rangle &= \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} \\ &= \begin{pmatrix}\frac{1}{\sqrt{2}}\\ -\frac{1}{\sqrt{2}}\end{pmatrix} \\ &= \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle. \end{aligned}$$

This is analogous to the "fair flip" matrix for probabilistic computing.

2.3 Quantum Computing Trick #1

One of the problems is that many of our operations are non-unitary/irreversible, so how can we use them? With the help of a helper bit b, we may define $U_f\colon \{0,1\}^2 \to \{0,1\}^2$ by

$$U_f(x,b) = (x, b \oplus f(x)),$$

where $\oplus$ is the XOR operator. Via some computation, we find that $(U_f \circ U_f)(x,b) = (x,b)$, so we have successfully made our operation reversible.

This allows us to run the quantum computation for a while. We can then stop it and reverse a few steps to help debug a program.

3 Lecture 3

3.1 Math Definitions

For the purposes of this class, a Hilbert space is a complex vector space with an inner product. We define our inner product to be

$$\left\langle \begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix} \middle| \begin{pmatrix}\beta_1\\ \beta_2\end{pmatrix} \right\rangle = \alpha_1^*\beta_1 + \alpha_2^*\beta_2 = \begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix}^\dagger \begin{pmatrix}\beta_1\\ \beta_2\end{pmatrix},$$

where $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \mathbb{C}$. We are also going to use Dirac notation:

$$|\psi\rangle = \begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix} \quad \text{and} \quad |\varphi\rangle = \begin{pmatrix}\beta_1\\ \beta_2\end{pmatrix}.$$

Furthermore, $|\psi\rangle^\dagger = \langle\psi|$. We also use the following notations:

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$
$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).$$

Then we have $H|0\rangle = |+\rangle$ and $H|1\rangle = |-\rangle$.

> **Note.** Unitary matrices preserve inner products.

We also have a notion of an outer product, defined by

$$\begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix}\begin{pmatrix}\beta_1^* & \beta_2^*\end{pmatrix} = \begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix}\begin{pmatrix}\beta_1\\ \beta_2\end{pmatrix}^\dagger = \begin{pmatrix}\alpha_1\beta_1^* & \alpha_1\beta_2^*\\ \alpha_2\beta_1^* & \alpha_2\beta_2^*\end{pmatrix} = |\psi\rangle\langle\varphi|.$$

3.2 Change of Basis

Suppose we have two bases: $\{|0\rangle, |1\rangle\}$ and $\{|\psi\rangle, |\varphi\rangle\}$. Suppose we want to change a vector $|v\rangle$ from the former basis to the latter basis. Then we have

$$\begin{aligned}|v\rangle &= \langle\psi|v\rangle \cdot |\psi\rangle + \langle\varphi|v\rangle \cdot |\varphi\rangle\\ &= |\langle\psi|v\rangle|^2 + |\langle\varphi|v\rangle|^2.\end{aligned}$$

3.3 Multiple Qubits

Suppose we have the start state

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

Suppose we measured the first qubit to be zero. Then the new state is

$$\frac{1}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}(\alpha_{00}|00\rangle + \alpha_{01}|01\rangle) = \frac{|0\rangle}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}} \otimes (\alpha_{00}|0\rangle + \alpha_{01}|1\rangle).$$

3.4 Tensor Product

> **Definition.** *Tensor Product*
> We define the *tensor product* (for two vectors) to be
>
> $$\begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix} \otimes \begin{pmatrix}\beta_1^* & \beta_2^*\end{pmatrix} = \begin{pmatrix}\alpha_1\beta_1^* & \alpha_1\beta_2^*\\ \alpha_2\beta_1^* & \alpha_2\beta_2^*\end{pmatrix},$$

which is the same as the outer product. In general, we have

$$\begin{pmatrix} \alpha_{00} & \alpha_{01} \\ \alpha_{10} & \alpha_{11} \end{pmatrix} \otimes B = \begin{pmatrix} \alpha_{00}B & \alpha_{01}B \\ \alpha_{10}B & \alpha_{11}B \end{pmatrix}.$$

Using this definition, we have

$$\begin{aligned} |00\rangle &= |0\rangle \otimes |0\rangle \\ &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 \cdot 1 \\ 1 \cdot 0 \\ 0 \cdot 1 \\ 0 \cdot 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \end{aligned}$$

Note. If we have $|\psi\rangle$ where ψ is some binary string of length n, then we have that $|\psi\rangle$ is a 2^n-dimensional vector where the ψ^{th} element (in decimal, zero-indexed) is a 1 and all the rest are 0.

3.4.1 General Rules

- The tensor product is *not* commutative, but is associative
- $A \otimes (B + C) = A \otimes B + A \otimes C$
- $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$
- $(\alpha A) \otimes B = A \otimes (\alpha B) = \alpha(A \otimes B)$