1 Introduction

Quantum computing lies at the unique intersection of computer science, mathematics, and physics. It utilizes quantum mechanical laws to solve some complex problems potentially faster than classical computers. In this course, we shall take the laws of quantum mechanics for granted as our axiomatic "ground truth" and shall focus entirely on their **computational consequences** [NC10].

1.1 Common Computational Tasks

Let's first look at some common computational tasks and their complexities on classical computers.

1.1.1 Multiplication

The classical primary-school (grid) algorithm for multiplying two n-digit numbers performs all digit-wise products explicitly and therefore requires a total of $O(n^2)$ single-digit operations. This was improved by Karatsuba, who introduced a divide-and-conquer strategy that reduces the multiplication of two n-digit numbers to three multiplications of size $n/2$ (with some lower-order additions). Hence we get the recursion

$$T(n) = 3T(n/2) + O(n).$$

yielding an overall time complexity of $O\big(n^{\log_2 3}\big) \approx O\big(n^{1.58}\big)$.

The fastest known classical algorithms for multiplication go even further by utilizing the Fast Fourier Transform (FFT). For example, the Schönhage–Strassen algorithm for integer multiplication achieves a runtime of $O(n \log n \log\log n)$ using FFT-based convolutions.

1.1.2 Factorization

The simplest (and most direct) algorithm for factorizing an integer is *trial division*. It finds a nontrivial divisor by checking integers $d = 2, 3, 4, \ldots$ and stopping when $d^2 > N$. This bound works because if $N = ab$ with $a \leq b$, then $a \leq \sqrt{N}$, and if no divisor exists up to $\sqrt{N}$, then N is prime.

In the worst case, this algorithm requires $O(\sqrt{N})$ divisions, or a net $O(\sqrt{N})$ number of arithmetic operations. If the input size is n (where $n = \log_2 N =$ number of bits), then

$$\sqrt{N} = 2^{\frac{1}{2}\log_2 N} = 2^{n/2}.$$

Thus, trial division has a runtime of $O(2^{n/2})$ and is exponential in the bit length of the input. Hence, this algorithm quickly becomes infeasible for practical use as the input size increases.

The most efficient known classical algorithm for factorization is the *General Number Field Sieve* (GNFS). Its runtime is

$$\exp\left(\left(\frac{64}{9}\right)^{1/3} (\log N)^{1/3} (\log\log N)^{2/3}\right),$$

which is sub-exponential in N but still super-polynomial in the input size, making it once again infeasible for practical use when N is large.

RSA
This apparent hardness of factorizing large numbers especially semiprimes (number that is a product of two large primes)is what is utilized by the RSA public key cryptosystems, widely deployed in practice. It exploits the fact that given a huge semiprime N(where N = pq and both p and q are large prime numbers satisfying some sanity checks), it is hard to factorize the number and find p and q.

1.1.3 Unstructured Database Search Problem

In the unstructured database search problem, the idea is that given a bit string $a_1a_2...a_n$, suppose it is promised that exactly one position contains 1 and rest all are 0s. The goal is to find the unique position i for which $a_i = 1$.

The classical deterministic approach to this problem is the brute force search algorithm. It checks all n possibilities in the input and finds the correct i for which $a_i = 1$ in an overall runtime of $O(n)$. In the deterministic setting, it is easy to see that $\Omega(n)$ is a lower bound as well. Even if we allow classical randomized computation, it can be shown that we still have a complexity of $\Theta(n)$ for this problem.

1.2 Role of Quantum Computing

As discussed earlier, Quantum Computing utilizes the principles of quantum mechanics to implement algorithms that can solve specific complex problems better than classical computers. Two examples for this are as follows which we shall see in details in this course:

1. **Shor's Algorithm (1994):** Peter Shor discovered a quantum algorithm that can factor integers in **polynomial time**. This gives an exponential speedup than the best known classical counterpart.
    - *Implication:* If we can build a large-scale quantum computer, there is an "easy" way of factorizing the product of 2 large prime numbers effectively breaking the public-key RSA cryptography.
2. **Grover's Algorithm (1996):** Lov Grover discovered a way to solve the unstructured database search problem on a quantum computer in time $O(\sqrt{n})$, which can be proved to be the lower bound as well making its quantum complexity to be $\Theta(\sqrt{n})$, which is a quadratic speedup than its classical counterpart.
    - *Implication:* Its application also extends to a number of other problems, such as the Boolean Satisfiability (SAT) problem. The goal of this problem is to determine whether a Boolean formula with n variables has an assignment that makes it evaluate to true. A deterministic exhaustive search would require testing all 2^n possible assignments, whereas Grover's algorithm can find a satisfying assignment in approximately $2^{n/2}$ steps. Thus, although the runtime remains exponential, the algorithm achieves a quadratic reduction. Consequently, Grover's algorithm does not reduce NP-complete problems to polynomial time, but it effectively halves the exponent in their time complexity.

2 Probabilistic Computing:

Probabilistic computing comprises of randomized algorithms that, in addition to the input, also have access to a source of randomness like independent and unbiased coin tosses. Hence, due to the source of randomness, the algorithm may lead to different execution paths for the same inputs.

Randomized algorithms can usually be divided into two types: Monte Carlo algorithms and Las Vegas algorithms. Monte-Carlo algorithms are those that always terminate within a prescribed polynomial time bound but may return an incorrect answer with some bounded probability. Las Vegas algorithms on the other hand always produce a correct output, but their running time is a random variable whose expected value is usually polynomial.

2.1 Primality Testing

One of the primary problems where randomized algorithms prove to be useful is primality testing. The goal here is to identify whether the input number is prime or composite. As discussed earlier, deterministic trial division takes $O(2^{n/2})$ time for this.

The Miller–Rabin test, on the other hand, uses Monte Carlo primality testing to achieve a complexity of $O(n^2 \log n \log\log n)$, which is polynomial in the input size. Solovay–Strassen [SS77] is another randomized algorithm that achieves this task in a runtime of $O(k \log^3 n)$, where k is the number of tested bases. It was only very recently (2002) that a deterministic algorithm for primality testing was found. The AKS algorithm [AKS04] provably runs in an overall time of $O(\log^{12} N)$. Subsequent improvements under specific assumptions reduces this to $O(\log^6 N)$ [LP19] .

BPP = P
A very strong belief in the theoretical computation community is that **BPP = P**. This means that any problem that can be solved in polynomial time with a bounded error probability (using randomness) can also be solved in polynomial time deterministically. If true, this would formalize the idea that random bits are merely a convenience for designing faster algorithms, and not a fundamentally stronger computational resource. Nevertheless, this remains officially unproven.

3 Quantum Computing: The Two Laws and Mathematical Formalism

In the previous section, we saw that probabilistic computing is "Classical Computing + Randomness." Quantum computing introduces something richer: interference via superposition and entanglement. A high-level summary is:

$$\text{Quantum Computing} = \text{Classical Computing} + \text{Interference (Rotation)}.$$

While probabilistic algorithms introduce randomness via coin tosses, quantum computing uses this extra structure to manipulate information in ways classical computers cannot. This is the source of quantum speedups:

- **Polynomial Speedups:** For problems like unstructured search (Grover's Algorithm).
- **Exponential Speedups:** For problems like factoring (Shor's Algorithm). [NC10]

Since quantum states behave like **vectors** that can be rotated, to understand these algorithms, we must rely heavily on **Linear Algebra**.

3.1 From Bits to Qubits

A classical bit has a state $x \in \{0, 1\}$. A quantum bit, or **qubit**, also has two basis states, which we label using the standard "ket" notation:

$$|0\rangle \quad \text{and} \quad |1\rangle .$$

For now, treat $|0\rangle$ and $|1\rangle$ as abstract labels. However, intuition comes from physics, where objects can be in more than one state at the same time:

- An **electron** may have two energy levels, labeled 0 (ground) and 1 (excited).
- A **photon** may have polarization that is Horizontal or Vertical.

If we label Horizontal as $|0\rangle$ and Vertical as $|1\rangle$, then before measurement, the photon may be in a superposition of both. After measurement, we observe only one outcome.

3.2 Law 1: Superposition

The first fundamental difference is that a qubit can be in a state that is a linear combination of the basis states.

Definition 3.1 (Quantum Mechanics Law 1: Superposition)**.** *If a quantum particle can be in one of the 2 basis states $|0\rangle$ and $|1\rangle$, then it can also be in a superposition state:*

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

*where $\alpha, \beta \in \mathbb{C}$ are complex numbers called **amplitudes**. These amplitudes must satisfy the **normalization condition**:*

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

3.2.1 Geometric Intuition (The Unit Circle)

If we restrict ourselves to **real** amplitudes ($\alpha, \beta \in \mathbb{R}$), the condition $\alpha^2 + \beta^2 = 1$ describes a circle of radius 1. Thus, we can visualize the state of a qubit as a point on the unit circle in $\mathbb{R}^2$.

Example 3.2. *Consider the state $|\psi\rangle = 0.8 |0\rangle - 0.6 |1\rangle$. Check normalization:*

$$|0.8|^2 + |-0.6|^2 = 0.64 + 0.36 = 1.$$

Geometrically, this corresponds to the point $(0.8, -0.6)$ on the unit circle.

Remark (Complex Amplitudes)**.** *In the general case, α and β are complex numbers $(a + ib)$. The magnitude is defined as $|\alpha| = \sqrt{a^2 + b^2}$. For example, if $|\psi\rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{i}{\sqrt{2}} |1\rangle$, then $|\frac{i}{\sqrt{2}}|^2 = \frac{1}{2}$.*

3.3 Law 2: Measurement

We cannot "see" the amplitudes α and β directly. We can only extract information via measurement.

Definition 3.3 (Quantum Mechanics Law 2: Measurement)**.** *Given a state $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$, if we measure it in the standard basis:*

1. *We observe outcome* $|0\rangle$ *with probability* $P(0) = |\alpha|^2$.
2. *We observe outcome* $|1\rangle$ *with probability* $P(1) = |\beta|^2$.

Crucially, after the measurement, the state ***collapses*** *to the observed outcome.*

Example 3.4. *Suppose we have the state* $|\psi\rangle = 0.8\,|0\rangle + 0.6\,|1\rangle$.

- ***Probability:*** *We see* $|0\rangle$ *with probability* 0.64*, and* $|1\rangle$ *with probability* 0.36.
- ***Collapse:*** *If we see* $|0\rangle$*, the state becomes* $|0\rangle$*. A second measurement will yield* $|0\rangle$ *with 100% probability.*

This is the key distinction from simply "revealing" a hidden coin toss. In the quantum world, the measurement forces the system to choose a reality.

3.4 The Mathematical Formalism: Qubits as Vectors

So far, we have used $|0\rangle$ and $|1\rangle$ as abstract labels. Mathematically, a quantum state is a vector in a complex vector space $\mathbb{C}^d$. For a single qubit, the dimension is $d = 2$. We identify the basis states with the standard basis vectors:

$$|0\rangle = e_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad |1\rangle = e_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Consequently, a general superposition $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ is simply a column vector:

$$|\psi\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \in \mathbb{C}^2.$$

3.5 Beyond 2 States: Qudits and General Laws

There is nothing special about having exactly two states.

- A **qutrit** ($d = 3$) has basis states $|0\rangle, |1\rangle, |2\rangle$.
- A general **qudit** (d) has basis states $|0\rangle, |1\rangle, \ldots, |d-1\rangle$.

In this general setting, the state is a vector in $\mathbb{C}^d$:

$$|\psi\rangle = \alpha_0\,|0\rangle + \cdots + \alpha_{d-1}\,|d-1\rangle \iff \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{d-1} \end{bmatrix} \in \mathbb{C}^d.$$

Each basis state $|i\rangle$ is the standard basis vector e_i (a column with a 1 in the i-th position and 0 elsewhere).

We can now restate our two fundamental laws in their full generality.

Definition 3.5 (Law 1 in dimension d: State)**.** *A general d-dimensional state is a superposition:*

$$|\psi\rangle = \sum_{i=0}^{d-1} \alpha_i\,|i\rangle, \quad \textit{subject to} \quad \sum_{i=0}^{d-1} |\alpha_i|^2 = 1.$$

Definition 3.6 (Law 2 in dimension d: Measurement)**.** *Measuring* $|\psi\rangle = \sum \alpha_i\,|i\rangle$ *outputs outcome i with probability* $|\alpha_i|^2$*. Immediately after measurement, the state collapses to* $|i\rangle$.

3.5.1 Example: The 4-dimensional system ($d = 4$)

A common example is $d = 4$. We can view this as a single system with 4 levels ($|0\rangle, |1\rangle, |2\rangle, |3\rangle$), or deeply importantly, as a system of **two qubits**. If we have two qubits, the combined system has $2 \times 2 = 4$ basis states:

$$|00\rangle, |01\rangle, |10\rangle, |11\rangle.$$

Here, $|01\rangle$ means "First qubit is 0, Second qubit is 1." This scaling (2^n dimension for n qubits) is the source of quantum computing's massive state space.

References

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. Primes is in p. *Annals of Mathematics*, 160(2):781–793, 2004. 3

[LP19] Hendrik W. Lenstra and Carl Pomerance. Primality testing with gaussian periods. *Journal of the European Mathematical Society*, 21(4):1229–1269, 2019. Previously cited as a 2005 preprint; establishes the $\tilde{O}(\log^6 n)$ deterministic bound. 3

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition edition, 2010. 1, 3

[SS77] Robert Solovay and Volker Strassen. A fast monte-carlo test for primality. *SIAM Journal on Computing*, 6(1):84–85, 1977. See also Erratum: SIAM J. Comput., 7(1), 118 (1978). 3