Lecture 1: Introduction to Quantum Computing

Rajat Mittal

IIT Kanpur

Whenever the word *Quantum Computing* is uttered in public, there are many reactions. The first one is of surprise, mostly pleasant, and then there are confused looks. Mostly we have questions like: what is quantum computing and why quantum computing; sometimes there are questions about the feasibility of a quantum computer too. Our aim in this lecture is to answer most of the basic questions people ask on hearing the term Quantum Computing.

1 What is quantum computing?

This course is about the theory of quantum computation, i.e., theoretical concepts needed to perform computation using quantum systems. These quantum systems follow the rules of quantum mechanics. We want to use these rules of quantum mechanics to compute the solution of problems. In other words, the task is to encode our computation (problems+solutions) in a quantum mechanical systems.

Before we explain the meaning of the previous paragraph, let us ask a more basic question.

Exercise 1. What is computation?

In a very informal manner, computation is a calculation done by a physical device (e.g., a *computer*) in an automated manner. The main idea is: instead of doing calculations on our own, we can use some physical device to do the calculations for us. You probably never worried about how a calculator (or computer) did the calculations for you; though, if we want to *improve* these devices, it is better to understand this question.

The computer is able to compute because of the encoding of the calculations using laws of nature (the ones you learnt in physics) which govern the working of these physical systems. In other words, because of the properties of current and voltage, we can implement AND, OR and NOT gates; these gates provide basic building blocks of a computer.

This fundamental idea, any physical law/process can potentially be used to do computation, led to the origin of quantum computation. An example will help in understanding this deep idea.

Suppose there is an imaginary insect, named Ro, which might have a ring on its body. Whether Ro will have a ring or not, it depends on its parents. Specifically, Ro will have a ring on its body if and only if one of its parents has a ring.

Following the idea in the preceding paragraphs, we can use Ro to create the OR gate; in other words, we can compute OR function using insect Ro.

Exercise 2. How?

If we need to take OR or of two bits b_1 and b_2, then create two Ro insects such that they have rings according to the value of b_i (insect has ring iff $b_i = 1$). Whenever these two Ros mate with each other, the offspring has a ring if and only if $OR(b_1, b_2)$ is 1. So, just observe the ring on the offspring and we will get our answer.

Since we are anyway creating an imaginary insect, let us assume that there is an injection, which changes whether the insect has ring or not. That means, an insect with ring will not have a ring after it gets an injection; on the other hand, an insect without a ring will get a ring after injection. This will create a NOT gate.

Exercise 3. What computation does Fig. 1 accomplish?

We are now ready to answer our original question,

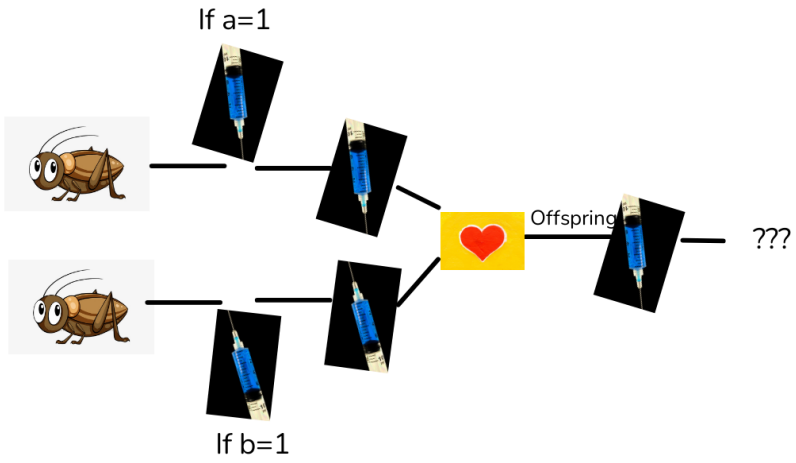

Fig. 1. Computation using insects

Exercise 4. what is quantum computing?

When computation is done using the principles of quantum mechanics, it is called quantum computing. In other words if the computation can be done on a physical device obeying quantum mechanical laws then it is called quantum computing. We will be looking at the potential and limits of this quantum model of computation.

Exercise 5. Everything in the world obeys quantum mechanics, then isn't every computation a quantum computation?

We know that classical and quantum laws behave similarly for a large scale system. It is only in the very small systems where peculiarities of quantum mechanics are visible. It is understood that quantum computers are useful (advantageous) only when these peculiarities can be exploited to do computation. So, we will call a system quantum computer, if it uses these distinguishing (*weird*) properties of quantum mechanics (like superposition and entanglement).

2 Why do we need quantum computing?

Church Turing hypothesis is one of the most fundamental assertion in computability theory. It says that any function calculable by a machine can also be computed by a Turing machine. In other words, if a machine can run an algorithm to do a task then there exist another algorithm to do the same task on a Turing machine.

Exercise 6. Can quantum computers break Church Turing hypothesis?

Sadly, that is not possible. It is known that any computational task which can be done on a quantum computer, can also be simulated on a classical computer. This means that anything we can do on a quantum computer can also be done/simulated by a classical computer. Then why do we need quantum computer? Succinctly, because quantum computers might be much faster.

The Church-Turing hypothesis has a stronger version, called *Strong Church Turing hypothesis*. It states that any algorithm, which can be performed on any machine, can be *efficiently* simulated using a Turing machine.

Strong Church-Turing hypothesis has worked for a long time. Still, this hypothesis might face challenges from different models of computation. In other words, it is possible that some weird physical law might compute things much faster than a classical Turing machine. Since we believe that quantum mechanics is

the theory of everything around us, it makes sense to have a quantum Turing machine instead of a classical Turing machine.

David Deutsch gave the concept of a quantum Turing machine. We don't have any algorithm/task on a quantum computer which violates the Strong Church-Turing hypothesis, but there are examples where the quantum algorithm works much more efficiently than the best known classical algorithm. Though, there is a possibility that we can find an efficient classical algorithm for these problems in future.

Exercise 7. Why do we believe that quantum computers can be faster than the normal classical computer?

Quantum mechanics has strange properties of superposition and entanglement. Superposition basically says that a quantum system can not only have classical states, but it can also exist in a *mixed* state (superposition) of these classical states. Because of the way states are defined in quantum mechanics, we also get the property of entanglement. Informally, the state of two systems cannot be described as the product of the state of one system times the state of other system.

It is natural to study these properties and their impact on computation. It seems reasonable that these properties can give rise to faster computation; in other words, quantum computing can be more resource-efficient than their classical counterparts. Last 30 years of research has given lot of evidences for the previous statement.

A brief history of ~~time~~ quantum computing: The history of quantum computing starts from the idea in 1970-80's that laws of quantum mechanics could potentially help in computation. The idea is generally attributed to Feynman and Yuri Manin. Feynman was particularly interested to simulate quantum physics and chemistry using a quantum computer. Paul Benioff laid the theoretical foundation of the idea of a quantum computer.

The starting years of this field (80's) were devoted to coming up with problems (seemingly contrived) which gave a "quantum advantage" (the *resources* required to solve the problems were lesser on a quantum computer). This lead to algorithms like Deutsch, Deutsch-Jozsa and Bernstein-Vazirani.

Arguably, the biggest results in the field of quantum algorithms came in 1990's, Grover and Shor's algorithm. These algorithms solved two very natural classical computer science problems (you would study them in the first course on programming with high probability) with a quantum advantage. Grover's algorithm gave a quadratic speedup (NOT exponential) for searching in an unstructured list. The importance of this result lies in the fact that search is utilized in almost all areas. Shor's algorithm gave a polynomial time algorithm for factoring an integer. We still don't know of a classical polynomial time algorithm, hence factoring is a candidate for exponential speedup on a quantum computer.

Later, these developments were further generalized in various ways leading to algorithms based on quantum Fourier transform and quantum walks. A "recent" important algorithm for solving linear equations under some special conditions was developed by Harrow, Hassidim and Lloyd in 2008 (popularly known as HHL algorithm). In last few years there have been a few important algorithms showing quantum advantage; for example Yamakawa-Zhandry result on random oracles and DQI algorithm by researchers from Google. We will not be able to cover these recent results, though they might be covered through projects.

Outline of the course: We are now ready to answer the question, what will we be doing in this course?

The first part of this course will cover the basics required to understand quantum computing. We will mostly be covering linear algebra: vector spaces, bases, rank and properties of some special class of matrices. We will introduce the postulates of quantum mechanics (thankfully that is all we will need from physics) as we finish the background of linear algebra required for them. With this knowledge, we will already be prepared to see some applications of quantum computing.

The next part will be focussed on introducing the model of quantum computation (as well as computation). Most of the algorithms in quantum computing are described in the circuit world (as opposed to Turing machine). We will describe this formalism and see some examples. The final part of the course will focus on algorithms which have shown that quantum computers can potentially be better than a classical computer.

You have already heard there names, Grover search and Shor's factorization algorithm. In general, the focus of this course will be on quantum algorithms. Another topic we will cover is to see for what problems quantum advantage is possible, we will study this question from the perspective of query model.

If time permits, the later part of the course will cover quantum walks. Unfortunately, we will not be able to cover quantum information theory and quantum cryptography, which are deep and very interesting topics in itself. Though, this course will create the background to learn these topics. We strongly encourage you to read and discuss these topics.

How to build a quantum computer: Before finishing the introduction, let us talk about the feasibility of a quantum computer. Various doubts have been expressed in building a quantum computer, and some researchers believe that it is not possible to construct one. Till this point, there is no fundamental barrier in making a quantum computer. There are obstacles, stability and scalability being very important, but intensive research has been going on to overcome these barriers.

Scientists have been trying to build quantum computers from a long time, to realize quantum computation on a physical device using various techniques. We do not have a desktop quantum computer but there are quantum computers on few qubits and many protocols have been experimentally realized. We will not be covering these areas; it is better to talk to a physicist to understand the complications in making a quantum computer. Again, students are encouraged to learn more about these topics on their own.

As a side note, we will mostly be concerned with the theory of quantum computation and information. It is important on its own, without worrying about a practical quantum computer. There have been many examples of results in classical computing, derived by the intuition gathered from quantum computing.

3 Quantum weirdness: Mach-Zehnder interferometer

Let us look at one (out of many) startling differences between the theory of quantum mechanics and our usual understanding of classical world. This example will help us in illustrating that quantum computer is simply not a probabilistic computer.

Suppose you have a photon gun and a beam splitter, look at Figure 2.

As the name suggests, a beam splitter will split a photon beam (coming from photon gun) into two parts, say horizontal and vertical. We can put detectors to check if the photon goes to the horizontal part or vertical part.

When multiple photons are fired by the photon gun, we observe that each detector clicks half the time. Sometimes the beam splitter lets the photon go (refract), sometimes it returns it (reflect). A skeptic might say, there is nothing unusual here, we can describe this by saying that the splitter reflects with probability $1/2$ and refracts with probability $1/2$. We already know how to do probabilistic computation, there does not seem to be anything new here.

A detour on randomized algorithms: In randomized/probabilistic algorithms, the input to the algorithm is the usual input (say x) and a random string (say $r \in \{0,1\}^t$). The output of the algorithm depends on both x as well as r. There are various ways to capture the performance of the algorithm,

- the expected cost (over r) is small and the algorithm answers correctly on all r,
- or the algorithm gives the correct answer on most of the r's but the cost is worst case over all r's.

Notice that we are interested in the behavior of the algorithm on input x; in randomized case, it means the cumulative behavior over all r's. Our state of the algorithms should capture this cumulative behavior and not just the behavior on each r individually. We will study these definitions formally and take concrete examples in the later part of the course.

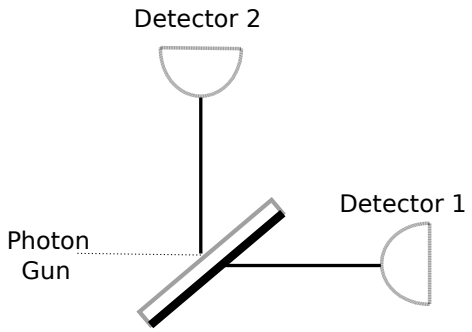

Fig. 2. A beam splitter and two detectors

Back to Mach-Zehnder: We make the experiment more complicated, and use another beam splitter and two mirrors (which always reflect). Notice the difference between a beam splitter and a mirror.

According to our probabilistic understanding, both beam splitters reflect with probability 1/2, we expect both detectors to light up 50% of the time. Instead, when the experiment is actually done, only detector 1 lights up; we never see a photon on detector 2!

Exercise 8. Read the Wikipedia article about Mach-Zehnder Interferometer to understand this phenomenon from the perspective of quantum mechanics.

There are two important observations here. The photon could go through two different paths, upper one or lower one.

- If we block one of the paths (remove one of the mirrors or place an obstruction), both detectors start lighting up. This is very counter-intuitive; we block a source of photon but that results in the increase in intensity at one of the detectors. This hints at cancellation of intensity from two different paths.
- The way to explain this phenomenon is that the light going to detector 1 has a constructive interference and the one going to detector 2 has a destructive interference because of a phase difference. Notice that the phenomenon remains even if we send one single photon (it behaves like a wave). In some sense, the photon passes through both the paths simultaneously.

This phenomenon can't be explained using probability distribution. It required us to give a new mathematical theory of these photons and their interaction with the beam splitter/ mirror. The theory is called quantum mechanics; hopefully, I have convinced you that it is quite different from the classical physics seen before.

We expect this new theory to give us more computational firepower. Like we cancelled intensity, if we can cancel unwanted computational paths, we can achieve advantage in computing. You will see numerous examples in this course, proving that such firepower is present.

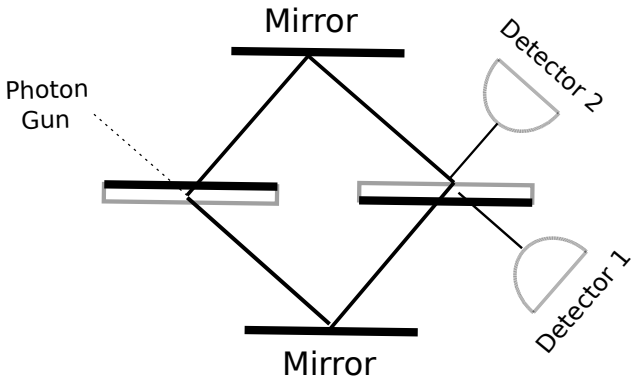

Fig. 3. Mach-Zehnder interferometer: Black edge is the reflecting edge in both beam splitter as well as the mirror. Notice that the "up" beam has to go through one more reflection.

There are many other cool experiments to show the power of quantum mechanics, one of them is called *Young's double slit experiment*. You can even see a videos Young's double slit experiment and Mach-Zehnder interferometer done at IIT Kanpur by Prof. Anand Kumar Jha of Quantum Optics and Entanglement lab ([1] and [2, 50 mins]).

3.1 Elitzur Vaidman bomb tester:

A very interesting application of Mach-Zehnder interferometer is the Elitzur-Vaidman bomb tester. In this thought experiment, we have a bomb which can be triggered by a photon. There are two possibilities, either the bomb is a dud (the photon will pass through it without doing anything) or it is a real bomb (it will explode by absorbing the photon). We would like to test whether the bomb is a dud or not, without exploding them (at least not explode them with high probability).

Notice the setup in Figure 4. We have placed the bomb at one of paths in the Mach-Zehnder interferometer. Let us check the two possibilities.

- Bomb is dud: In this case, the device resembles the traditional Mach-Zehnder interferometer. We will always observe the photon at Detector 1.
- Bomb is real: The photon can pass through upper path or the lower path. If it passes through the bomb, the bomb will explode and the superposition is destroyed. This will happen half the time. In the remaining half, the light beam will go to second beam-splitter and will be detected at detector 1 or 2 with equal probability. That means, if we detect a photon at detector 2, the bomb was real and it didn't explode (this happens with probability 1/4).

So, to conclude, if the bomb is real, it explodes with probability 1/2. With probability 1/4 we incorrectly call it a dud. With probability 1/4 we get the magical result, we correctly test the bomb without exploding it. This probability can be made arbitrarily close to 1 by modifying the setup (we will see this later, impatient readers can read the wiki).

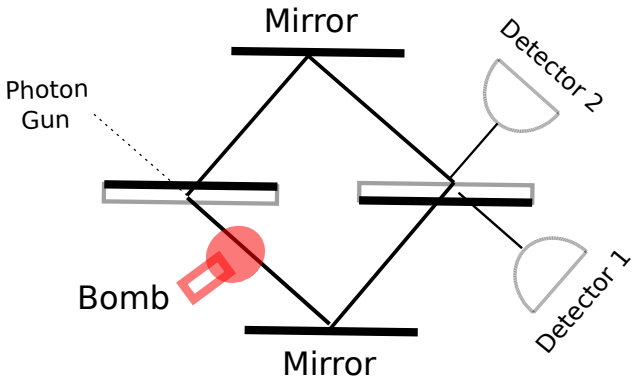

Fig. 4. Elitzur Vaidman bomb tester: how to test a bomb without exploding it.

Both, Mach-Zehnder and Elitzur-Vaidman, can be clearly explained using the formalism of quantum mechanics. We will do so in this course once you have learnt that formalism.

4 What is next in our journey?

The previous experiment gives evidence that quantum mechanics gives rise to many surprising effects and properties (some might call them weird). If you want to use these properties, it might seem that you need to be an expert in quantum mechanics. Fortunately, we can capture most of the quantum mechanics needed in a few postulates.

Exercise 9. What do these postulates describe?

The postulates describe the basic guidelines on which any quantum system works. To understand how any computational system works, we need to understand following things.

- How do we store information or data? This is done using bits in a classical computer. A bit can be in state $\{0, 1\}$. All numbers (and string, and images, and videos) can be written as a string of these bits. What will be the analogous concept, *quantum bits (qubits)*, in a quantum computer?
- What operations are permitted over this data? We use building blocks like *OR* and *AND* gates, to create complex circuits which manipulate data. How can we manipulate quantum bits?
- How do we obtain the final result of the computation? This might not seem like a non-trivial question in the classical world, just observe the bits at the end of the computation. Due to the nature of quantum mechanics, this question becomes more complicated for a quantum computer.
- Another trivial question which becomes relevant in quantum computing is: how do we represent multiple systems? For the classical case of two bits, you will just write them next to each other to represent two bits. Surprisingly, that is not enough for two quantum bits.

In the next few lectures we will learn concepts in linear algebra, which will allow us to give a precise mathematical answer to above questions using postulates of quantum mechanics. Not just that, we will also be able to explain *quantum effects*, like the one in the previous section, by a clean mathematical description. As a computer scientist, we can (almost) forget about quantum mechanics after understanding these postulates, and just focus on the computing part of quantum computing.

5 Assignment

Exercise 10. Read about randomized computing. Is it different from quantum computing?

Exercise 11. Read about DNA computing.

Exercise 12. If we take the probabilistic interpretation for Mach-Zehnder apparatus, what is the probability of reaching detector 1 when the first beam splitter reflects?

Exercise 13. Read about entanglement and Bell's inequality.

References

1. A. K. Jha. Young's double-slit single photon experiment, 2021. `https://www.youtube.com/watch?v=UgBvOhsFvKk`.
2. A. K. Jha. Live demonstration of quantum erasure and interference, 2023. `https://www.youtube.com/watch?v=5fKeTy0tET8`.