**16**

**Quantum Computing**

Quantum technologies are a rapidly growing and interdisciplinary field of science and engineering. Until now, we've depended upon supercomputers to solve most problems, but **quantum computing** (**QC**) accelerates complex calculations exponentially, resulting in a very powerful supercomputer. As per quantum physics, a *quantum* is the smallest unit of any physical property and refers to the properties of atomic particles, such as electrons, neutrinos, and photons. QC can solve computational problems beyond classical computers' reach by utilizing the laws of quantum mechanics to process information in new ways.

QC is a new paradigm in computing that speeds up solutions to complex problems, giving not 10x performance improvement but 10**X**. This approach to computing could transform chemical engineering, material science, drug discovery, financial portfolio optimization, and machine learning.

QC is still in the research stage, and it may take some years before it gets commercialized. Quantum technology is complex but, in this chapter, you will learn QC concepts in a simplified way by exploring the following topics:

* The building blocks of quantum computers
* The working mechanism of quantum computers
* Quantum gates
* Quantum circuits
* Types of quantum computers
* Quantum computing in the cloud

By the end of this chapter, you will understand the foundations of QC. You will know about the workings of quantum computers, including their building blocks, types, and various use cases. You will also learn about the logic behind quantum algorithms, such as quantum gates and circuits, and how quantum computers work behind the scenes.

**The building blocks of quantum computers**

The basic building block of a quantum computer is the **qubit**. Qubits in QC are similar to bits in classical computing, but they behave very differently. Let's learn more about qubits.

**Qubits**

Classical bits can hold only a position of 0 or 1, but qubits can hold positions between 0 and 1, and multiple positions at once. Qubits are quantum systems that scientists and engineers can control. Some examples of qubits are atoms, molecules, and photons.

Quantum states are represented by the little flags "https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_001.png", called **Dirac notation**. A single classical bit is denoted by 0 or 1, while a single quantum bit (qubit) is denoted by complex linear combinations of https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_002.png and https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_003.png. Qubits are two-dimensional vectors with complex coefficients, as shown below:

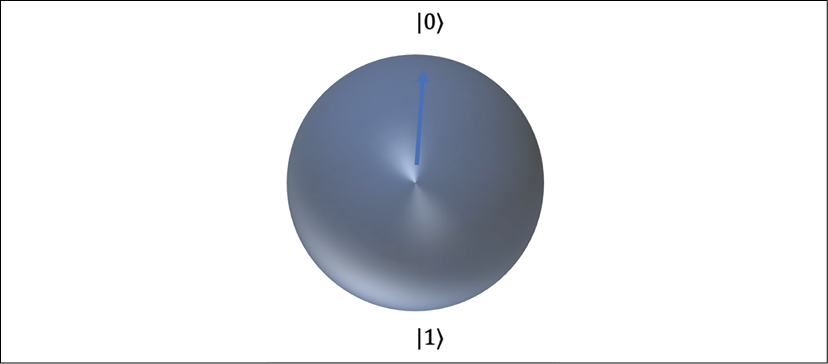


Figure 16.1: Bloch sphere – abstract representation of a qubit

In the preceding diagram, a qubit is represented using a **Bloch sphere**. A Bloch sphere is a representation of the pure state space of a qubit, named after the physicist Felix Bloch. The blue arrow indicates the state of the qubit at any given time. You can see two states represented in the diagram with https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_004.png and https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_003.png. When represented as two-dimensional vectors, the values of https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_006.png and https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_003.png are:

A picture containing text, clock, watch

Description automatically generated

The possible states of two qubits can be basis vectors https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_008.png, https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_009.png, https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_010.png, https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0111.png, in addition to superpositions, as opposed to the possible states of two classical bits, which are 00 or 01 or 10 or 11.

Quantum computers utilize the unique behavior of quantum physics, such as superposition and entanglement. Let's understand these concepts.

**Superposition**

To simplify, let's take an example of classical fair coin-tossing to understand the various concepts of QC. Tossing a coin gives you a head or a tail with a 50-50 chance, as with a classical computer. 1 bit has two possible values—we can assume heads being 1 and tails being 0. Quantum computers use qubits to process information, and 1 qubit represents one quantum state, which could be any state while a coin is being tossed. Imagine if you were able to look at a coin and see both heads and tails simultaneously, as well as every state in between: the coin would be in superposition.

A qubit itself isn't very useful. However, we can create vast computational spaces by creating many spaces and connecting them in a state called superposition. In superposition, quantum particles represent a combination of all possible states. They fluctuate until they're observed and measured. In this case, if you flip 50 coins simultaneously, a qubit can be in a superposition of heads and tails, and 50 qubits can be in any superposition of 2**50** states. Now you can see that a small quantum computer with just 50 qubits can store one million billion complex numbers derived from 2**50** ~ 1,000,000,000,000,000.

Now that you have learned about qubits, let's see how two qubits work together through quantum entanglement.

**Entanglement**

Entanglement is the ability of quantum particles to correlate with each other. Quantum computers can calculate exponentially more information and solve more complicated problems by entangling more qubits in a system. Entangled qubits form a single system and, by relating to each other, draw a correlation and get an outcome. You can use the measurements from one qubit to draw conclusions about the others.

Quantum entanglement allows qubits, which behave randomly, to be perfectly correlated with each other. Let's understand entangling from a programming point of view in the table below by comparing the properties of classical bits to qubits.

|  |  |  |
| --- | --- | --- |
|  | Classical Computer: Copying X Bit to Y Bit | Quantum Computer: Entangling X Qubit and Y Qubit |
| **Correlation** | X bit and Y bit are uncorrelated after copying | X qubit and Y qubit correlate; measuring X affects Y instantaneously |
| **Referencing** | May assign by reference on same data so X bit and Y bit may point to the same data | Entangled qubits exist individually, but they correlate |
| **Reversibility** | It's irreversible—the reverse of an operation like copying back X to Y would destroy Y | It's reversible—the entangled qubits X and Y can be unentangled |
| **Correction** | For error correction, bits can be restored from a previous copy | Quantum error correction uses many entangled qubits |

As you can see in the above table, entangling is not copying. Using quantum algorithms and quantum entanglement, specific complex problems can be solved more efficiently than with classical computers as you can relate multiple qubits to draw a conclusion and utilize them to store large values. An array of qubits can use superposition to represent 2^64 possible values at a time, which allows a quantum computer to solve complex problems that are impossible for standard computers.

Now you understand the building blocks of QC. Let's explore more of their working mechanism and what makes them so fast.

**The working mechanism of quantum computers**

You learned about qubits in the previous section. Now, to build quantum computers, physical qubits are built in a lab, and the quantum computer has an area that houses the qubits.

The unit that houses the qubits is kept at a temperature just above absolute zero to maximize the qubits' coherence and reduce interference. The low temperature helps to stabilize and control the qubits. A vacuum chamber can also be used to help minimize vibrations and stabilize the qubits. Signals can be sent to the qubits by using various methods, including microwaves, lasers, and voltage.

You must be wondering how quantum computers run so fast. A quantum computer runs so fast because it tries all answers to a problem in parallel. Is that true? Well, not really, not in the sense that multiple calculations are all happening in parallel universes. That said, qubits can hold 0 and 1 values at the same time using superposition, and let them influence each other with entanglement. This opens up new possibilities for building clever quantum algorithms to speed up computation.

To make quantum computers faster, one promising quantum algorithm that's used is called Grover's search. Suppose you need to find one item from a list of N items. On a classical computer, you'd have to check N/2 items on average, and in the worst case, you would need to check all N items. Using Grover's search on a quantum computer, you can find the item after checking roughly √N of them. This represents a remarkable increase in processing efficiency and time saved. For example, if you wanted to find one item in a list of 1 trillion, and each item took 1 microsecond to check, on a classical supercomputer the search might take over a week, but a quantum computer accomplishes the same task in a matter of seconds.

Let's use the analogy of making music. When musicians play beautiful music, they make it by styling the sound of a musical chord, the same way quantum computers work by manipulating the amplitudes of the state vector. To write a song, you figure out what notes you need and put them together into a musical arrangement; in the same way, to program a quantum computer, you arrange qubits into a quantum circuit and run the circuit. Finally, when you produce music, your band comes together and plays the musical instruments in a way that sounds good; in the same way, you set quantum circuits so that the answer given is the most probable one.

Compared to classical computers, quantum computers can create vast multidimensional spaces to represent very large problems. Quantum algorithms then find solutions in this space and translate them back into forms we can use and understand. Due to superposition, quantum interference in the intrinsic behavior of a qubit can influence the probability of it collapsing. Quantum computers are designed and built to reduce noise as much as possible and ensure the most accurate results. Quantum logic is built by various types of logical gates. Let's learn more details about building quantum logic.

**Quantum gates**

Whether classical or quantum, a logic gate is any physical structure or system that takes a set of binary inputs (1s and 0s or spin-up/spin-down electrons) and gives a single binary output: a 1 (that is, a spin-up electron), or one of two states of superposition. What governs the outcome is a *Boolean function*. You can think of a Boolean function as a rule for responding to Yes/No questions. The gates are combined into circuits and the circuits are combined into CPUs or other computational components.

To understand gates better, let's look at the Bloch sphere in 3D space, which represents the direction of a quantum particle, such as an electron's spin, along a corresponding axis. You learned about the Bloch sphere in the previous topic, *The building blocks of quantum computers*.

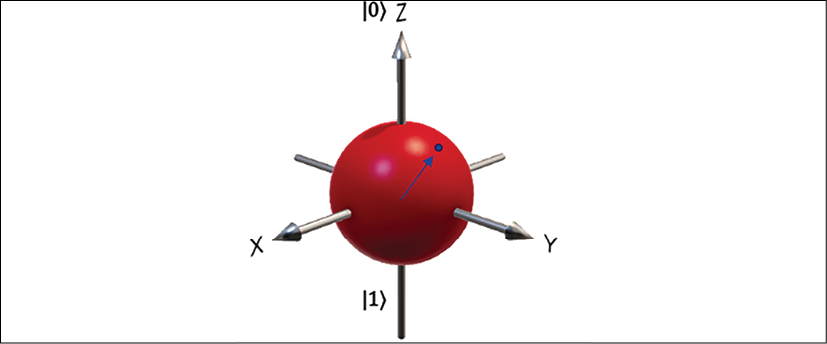


Figure 16.2: 3D representation of an electron's spin along with its axes

The preceding diagram represents a qubit in 3D space; qubits can be placed in a spin-up state along the X, Y, and Z axis using quantum gates. Quantum gates operate on qubits, as opposed to classical gates operating on 0 or 1 bits, which makes it possible for quantum gates to use *superposition* (other states between 0 and 1) and *entanglement (relating two qubits to drive an outcome)*.

Many types of gates are utilized to build a quantum circuit, and the simplest one is the single-qubit Pauli gate. Let's learn more about the Pauli gate.

**Pauli gates**

The Pauli gate is named after Wolfgang Pauli, and is based on the better-known Pauli matrices.

Pauli gates are very useful for calculating changes to the spin of a single electron. Today's quantum gates use electron spin as the favored property for a qubit. Pauli gates act on only one qubit at a time. This translates to simple 2 x 2 matrices with only 4 elements apiece. Pauli gates come in three types: Pauli-X, Pauli-Y, and Pauli-Z.

**Pauli-X Gate**: A Pauli-X gate can be related to a classical NOT gate and is often called *the quantum NOT* gate. The following denotes it:

A picture containing text, clock

Description automatically generated

Figure 16.3: Pauli-X gate

Now, If you take a linear state of a single qubit vector and pass it through a Pauli-X gate, it will flip to another side, for instance, from https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_002.png to https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_003.png and vice versa.

**Pauli-Y Gate**: This looks a lot like the X-gate, but with i (square root of -1) in place of the lower-left 1 and negative i in the upper-right corner. The following denotes it:

A picture containing text, clock

Description automatically generated

Figure 16.4: Pauli-Y gate

When a qubit in the form of an electron passes through a Y-gate, it spins to the *Y* axis in 3D space.

**Pauli-Z Gate**: This looks kind of like a mirror image of the X-gate, but with a negative sign in the lower right. The following denotes it:

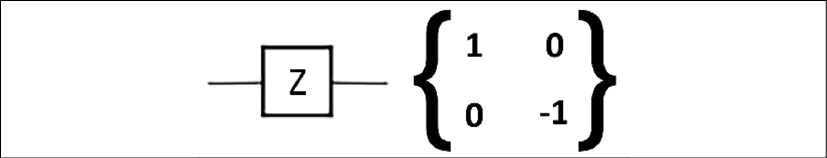


Figure 16.5: Pauli-Z gate

When a qubit in the form of an electron passes through a Z-gate, it flips to the *Z* axis in 3D space.

The Y-gate and the Z-gate also change the spin of a qubit electron. But let's look at another very important gate, Hadamard, which you will see in any quantum circuit.

**The Hadamard gate**

Hadamard is also known as the H-gate, and it shows up everywhere in QC. The Hadamard gate has the *quantum* capacity to transform a definite quantum state of a qubit, such as a superposition of *both* spin-up and spin-down simultaneously. The following denotes it:

A picture containing text, clock

Description automatically generated

Figure 16.6: Hadamard gate

Once you send an electron through an H-gate, it will become like a coin in mid-air; with 50/50 odds, it will end up heads or tails when it lands. The H-gate is very useful for performing the initial computation in a quantum program as it transforms *initialized* qubits back into their natural fluid state to leverage their full quantum powers. An H-gate maps Xhttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.pngZ and Zhttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.pngX as it operates between two states simultaneously, so an H-gate is required to make superpositions.

There are several more quantum gates. Let's explore some popular gates.

**Other quantum gates**

There are many other quantum gates you're bound to use. While some gates operate on single qubits, others operate on several qubits at a time. Let's look at some more gates that operate on a single qubit, which can be visualized as transformations around a Bloch sphere:

* **X-Gate**: Also referred to as a flip-bit, this performs https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_014.png rotation around the *X* axis, changing Zhttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.png-Z and back again if done twice.
* **Z-Gate**: Also referred to as a phase-flip, this performs https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_014.png rotation around the *Z*-axis, changing Xhttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.png-X and back again if done twice.
* **S-Gate**: This performs https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_016.png rotation around the *Z*-axis and maps Xhttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.pngY. This gate further extends the H-gate to make complex superpositions.
* **S'-Gate**: This is the inverse of the S-gate and maps Xhttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.png-Y. It performs -https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_017.png rotation around the *Z*-axis.
* **T-Gate**: This performs https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_018.png rotation around the *Z*-axis.
* **T'-Gate**: This is the inverse of the T-gate and performs -https://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_019.png rotation around the *Z*-axis.

Some of the other popular gates are the Toffoli gate, the Fredkin gate, the NOT-gate square root, the Deutsch gate, the swap gate (and the swap-gate square root), the Controlled-NOT gate (C-NOT), and other controlled gates. But we are not going to explore all the gates; we've covered the main ones above. You've learned about all the basic gates, so let's see how to put them together and build a quantum circuit.

**Quantum circuits**

Now you've learned about some of the fundamental quantum gates, so the next question is how to perform quantum logic. To build a quantum algorithm, you need to use well-controlled qubits for computing. Those qubits can interact and change states as desired. An algorithm is built by manipulating quantum states (vectors) using quantum gates (matrices). Let's look at the Boolean circuit in a classical computer built with three inputs.

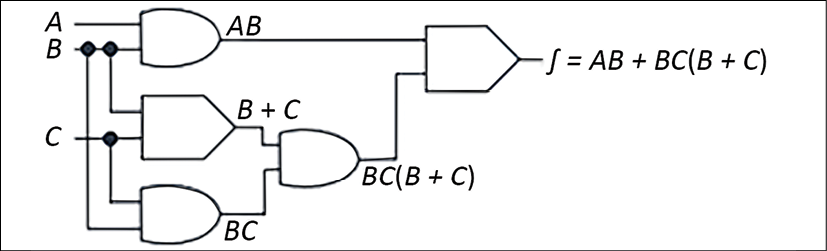


Figure 16.7: Operation in a Boolean circuit

In the preceding diagram, you have a combination of multiple Boolean gates where an AND gate is multiplying two inputs, resulting in A & Bhttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.pngAB and B & Chttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.pngBC. Further, an OR gate is adding two inputs, resulting in B & Chttps://learning.oreilly.com/api/v2/epubs/urn:orm:book:9781801816618/files/Images/B17764_16_0001.pngB + C. You can see that by roping these gates in a circuit, an algorithm has been built.

That is the way classic computers work. Now, let's take a look at the quantum circuit in the following diagram, which you can compare with the above Boolean circuit:

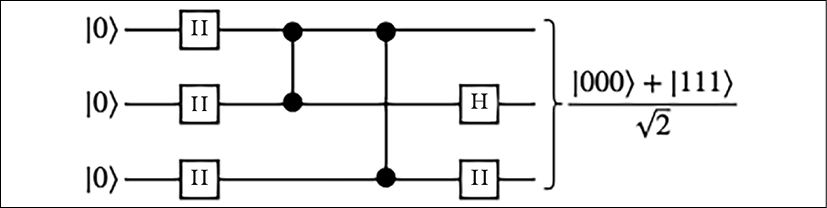


Diagram 16.8: Operation in a quantum circuit

In the preceding diagram, you pass three qubits with zero vector state. They pass through the H-gate and change the state to create a superposition over the course of some further steps (we're avoiding the details here, as it's beyond the scope of this chapter to explain the full complexities of quantum algorithms), resulting in one of two equally likely final measurement outcomes, [000] and [111], each over the square root of 2, as shown in *Figure 16.6*.

Basic quantum circuits consist of three basic modules:

1. **Initialize**: Prepare the initial state
2. **Execute**: Execute unitary gates on the qubits
3. **Measure**: Measure the qubits on some reference basis

Here, in the quantum circuit, the digital Boolean circuit problem is broken down into a sequence of a few basic building blocks: a set of universal gates mentioned in the previous section and an error correction process to achieve fault tolerance. There are growing numbers of quantum algorithms, such as:

* **Number theory**: Factoring, discrete log, subset-sum, and cryptanalysis
* **Optimization**: Constraint satisfaction, solving linear systems, solving ODEs, PDEs
* **Oracular**: Search, hidden subgroup, hidden shift, order finding
* **Simulation/approximation**: Knot invariants, QAOA, SDPs, direct simulation

We are not going to go into the details of algorithms as this is a very vast topic. These algorithms are generally use case-agnostic, and QC can be used in any application with one of many problems at its core. These algorithms will be helpful when we have a universal fault-tolerant quantum computer. In this section, we explained the basic building blocks of quantum computers to understand the future trends in the coming decade.

Quantum computers are at a very early stage and there is a lot to explore. You will only want to use such computers if your organization's problems require high-level calculations and the storage of large values beyond the limits of classical computers, such as calculating trillions of numbers at once.

You've looked at gate-based QC; another type of QC is available called quantum annealing. In **quantum annealing**, quantum operations are analog rather than gate-based digital operations. These are more complex and special-purpose optimization machines. Quantum annealing is used to encode a solution to an optimization problem by finding the ground state of a problem, or the zero-point energy of the system.

Let's get into the real world and see the types of quantum computer hardware that are available.

**Types of quantum computers**

As you learned in the previous section, there are two QC paradigms. The first is gate-based using quantum computers, with fewer, higher-quality qubits for general-purpose devices. Most quantum computers are gate-based, such as **Rigetti and IonQ**.

The second QC paradigm involves quantum annealers with many qubits built for special purposes. **D-Wave**-built quantum computers are based on quantum annealing. D-Wave provides quantum computers of up to 2,000 qubits. Let's look at some quantum computers based on the different types of particles used:

* **Trapped Ions**: The first quantum logic gate was demonstrated in 1995 using trapped atomic ions. It uses two internal states of ions, whose positions are controlled with electric fields in "ion traps" as their fundamental quantum element. The state of each ion can be changed using controlled microwave radiation. These pulses can be arranged to couple the states of two or more ions to create entanglement between the ions.
* **Rydberg Atoms**: This is similar to trapped-ions QC, but neutral atoms are used instead of charged ions for the qubits. Laser tweezers are used to keep the particles in place instead of electric fields. As with trapped-ions qubits, optical and microwave pulses are used for qubit manipulation. What makes this different compared to trapped ions is its potential for building multidimensional arrays.
* **Superconducting Qubits**: This is considered the forefront of qubit modalities for digital quantum computation and quantum annealing. Microwave radiation is used to manipulate the state.

Adjacent quantum particles can be electronically coupled together to create entangled states. They exhibit quantized energy levels due to quantized states of electronic charge or magnetic flux, for example.

As QC gains more traction, public cloud providers such as AWS, Google, Microsoft Azure, and IBM have started providing easy access to quantum computers through their cloud platforms. Let's learn about some real-life use cases where QC will be useful.

**Quantum computing in real life**

Quantum computers are collections of specialized quantum systems that can be systematically controlled over a period of time to perform a desired task. QC is not a replacement for current computers and may not solve problems where you don't need complex calculations. As an analogy, you can see light bulbs versus candlelight as equivalent to QC versus current computers. It doesn't matter how much advancement you put into candles; you cannot convert candles into light bulbs—they are entirely different technologies.

Let's take an example of choosing the seating arrangements for 15 people. At first glance, it may look straightforward, but if you calculate, there are more than 1.3 trillion (factorial of 15) possible ways to seat just 15 people. Imagine if you needed to solve this problem for 100 people: you would run out of memory and compute. Classical supercomputers don't have the working memory to hold the countless combinations of real-world problems. The way they are designed means they would take a long time, analyzing each combination one after another.

In the same vein, think about other real-world problems, like if Amazon wants to optimize drivers' routes across 100 cities to find the optimal route for quick deliveries. Another use case would be if a pharma company wants to build a drug repurposing platform by simulating molecules to understand drug interactions better. These problems are hard to solve using classical computer technology, due to the high number of calculations required.

Some other real-time use cases that can be solved by QC include:

* **Optimization**: Optimization can be a very complex process, involving finding the right solution among millions and billions of combinations. You can optimize your product quality, cost, and efficiency. Solving complex optimization through quantum algorithms helps us better manage complex problems, such as airplane gate assignment, package deliveries, traffic control, and energy storage.
* **Machine learning**: Machine learning is already helping business and science even with classical computers. However, training machine learning models has a relatively high cost, which has hindered the field's scope and capacity for development. Quantum computers can help speed up progress in this area with faster training and larger amounts of data.
* **Internet searches**: Internet searching is another area where you need to optimize and index trillions of pieces of information. QC can help to make searching faster for high or even unlimited amounts of data.
* **Simulation**: Quantum computers work exceptionally well for modeling other systems in their computation. They can handle the complex systems that would overwhelm classical computers. Examples of quantum systems that we can model include drug simulation, photosynthesis, superconductivity, and complex molecular formations.
* **Cryptography**: Classical cryptography using RSA depends upon the intractability of integer factorization or discrete logarithms. Encryption and decryption problems can be solved more efficiently using quantum computers.

QC works very differently to classical computers. As you now understand some QC use cases, let's learn more details about the building blocks of quantum computers.

**Quantum computing in the cloud**

Like any other technology, public cloud providers facilitate easy access to QC. Procuring quantum computers is very costly, and it's still at the very initial stage where you may not see a return on your investment in the near term. But you don't want to miss the bus; you want to get your foot in the door with this upcoming technology that has the potential to change the world as we see it today. The public cloud is the perfect place to access quantum computers and try out your use case.

Amazon Web Services provides a QC service called Amazon Braket. AWS doesn't have quantum computers, but they provide access to other gate-based and annealing-based quantum computers through their platform. AWS provides access to D-Wave's Advantage and 2,000 qubit computers, IonQ, and Rigetti quantum computers through their Braket platform. Amazon Braket provides a hardware-agnostic quantum platform with scalable circuit simulators in the AWS Cloud and end-to-end integration with AWS Cloud services. Similarly, Azure Quantum also operates through a partner network.

Google Quantum AI provides access to superconductor-based quantum computers based on the Sycamore processor with 54 qubits. It has a single qubit and double qubit-based circuit. Google provides a Python framework called Cirq to build and optimize quantum circuits. Their Weber quantum computer is 53 qubits.

IBM's quantum computer is based on superconductors. IBM uses superfluids to chill their superconductors to about a hundredth of a degree Celsius above absolute zero. After electrons are put through the superconductors, they pair up into something called Cooper pairs, which quantum tunnel through something called a Josephson junction. These are superconducting qubits. By firing photons at a qubit, we can control its behavior and get it to hold, change, and read out information. IBM creates vast computational spaces using programmable gates by creating many qubits and connecting them in a superposition state. Finally, quantum entanglement allows qubits, which behave randomly, to be perfectly correlated with each other. Using quantum algorithms that exploit quantum entanglement, specific complex problems can be solved more efficiently than on classical computers.

QC is a very complex and vast topic. We are still scratching the surface right now, and there are many more things to come in the future. It may take another decade to realize the full potential of quantum computers and build robust, fault-tolerant quantum computers. However, this chapter should have given you an idea of this potential next technological revolution and the possibilities of solving complex future problems that looked impossible until now, like space travel and human genome decoding.

**Summary**

QC has a lot of potential to solve complex problems that even the most powerful supercomputers cannot solve. Now the world has started to build quantum computers and algorithms, yet we are just scratching the surface, and it may take another 5 to 10 years before we start realizing the commercial value.

In this chapter, you learned about QC and some real-life use cases where QC can be advantageous. QC is not the answer to every problem, and it is only applicable to complex calculations where classical computers won't work.

You learned about the key building block of the quantum computer, the qubit, and how multiple qubits work together in a state of superposition and entanglement to solve a complex problem. You learned about the working mechanism of the quantum computer and understood why they are faster than classical computers.

To perform operations on qubits, you learned about QC gates, such as Pauli and Hadamard gates, which apply to single qubits, and other gates. You also learned about quantum annealing, which is a different paradigm based on analog rather than digital gates. Further more, you put together these gates and understood how QC circuits work compared to classical Boolean circuits.

Finally, you learned about various types of quantum computers and the technology behind them.

You also learned about quantum computer providers such as D-Wave, Rigetti, and IonQ. You learned about the Amazon Braket platform, which provides easy access to multiple quantum computers in a pay-as-you-go model. You learned about working on IBM quantum computers and Google quantum processors.

While, in this chapter, we looked far into the future, many legacy workload enterprises are working to modernize. With time, organizations tend to accumulate technology debt, and many legacy applications are sitting in the data center, creating costs and consuming resources. In the next chapter, you will learn about legacy application transformation and modernization. You will learn about the challenges with legacy systems and the techniques used to modernize them.