Quantum Computation and Complexity

Anonymous Author(s)
Stanford University

Abstract

This paper is a brief introduction to quantum computation and a survey of important results regarding quantum computation and quantum complexity theory. The paper is structured as follows. In Section 1, we give an overview of quantum mechanics and the building blocks of quantum computation. In Section 2, we look at examples of famous quantum algorithms that solve problems much faster than existing classical algorithms. In Section 3, we analyze the complexity class BQP, the class of problems that efficient quantum algorithms can solve. Finally, in Section 4, we introduce the class QMA, which is the quantum analogue of NP. Throughout the paper, we will only give high-level explanations of the theorem proofs and leave out the rigorous and messy details.

1 Introduction

In this section, we will give a brief introduction to quantum mechanics and some basic elements of quantum computation. Feel free to skip ahead if you are familiar with these topics, and refer to [3, 4] for more advanced materials.

1.1 Quantum Mechanics

The main postulate of quantum mechanics is that any isolated physical system has an associated **complex vector space** (state space), and the system is completely described as a **unit vector** (state vector) in the state space. The standard notation is the **Dirac notation**: a state vector is denoted as $|\psi\rangle$, its complex conjugate $|\psi\rangle^\dagger$ is denoted as $\langle\psi|$, and the inner product of two states $|\phi\rangle$ and $|\psi\rangle$ is $\langle\phi|\psi\rangle$. Usually, a vector $|\psi\rangle$ in a d-dimensional space is written as $|\psi\rangle = \sum_i^d a_i |e_i\rangle$, where $\{|e_i\rangle\}$ is some orthonormal basis in the state space, the a_i's are complex numbers, and $\sum_i^d |a_i|^2 = 1$.

Given a state $|\psi\rangle = \sum_i^d a_i |e_i\rangle$, we can make a **measurement**. The most simple case of a measurement is the measurement with respect to basis $\{|e_i\rangle\}$, where the outcome may be $\{1, \ldots, d\}$, and the probability of getting outcome i is $p(i) = |a_i|^2$. After the measurement and obtaining outcome i, the state "collapses" to $|\psi'\rangle = |e_i\rangle$. This is one of the weird properties of quantum mechanics. Given a $|\psi\rangle$, the measurement outcome is random, and after the measurement, the original state is destroyed, collapsing to one of the basis vectors.

Finally, in a closed physical system, a transformation of a state $|\psi\rangle$ is described by $|\psi'\rangle = U|\psi\rangle$, where U is a **unitary matrix** ($U^\dagger U = I$). Under any unitary transformation, $\langle\psi'|\psi'\rangle = \langle\psi|U^\dagger U|\psi\rangle = \langle\psi|\psi\rangle = 1$, which ensures that the new state is still a unit vector.

1.2 Qubits

Consider a 2-dimensional state space, and let $|0\rangle$ and $|1\rangle$ be an orthonormal basis for the state space. We can easily see that this is a direct analogy to a classical bit, which can take value 0 or 1. Any state

vector can be written as

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle = \begin{bmatrix} a \\ b \end{bmatrix} \tag{1}$$

We call this a quantum bit, or a **qubit**. If we make a measurement, we will get outcome 0 with probability $|a|^2$, and outcome 1 with probability $|b|^2$. Unlike a classical bit, a qubit can be "both 0 and 1". For example, if we measure a state $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, we will get 0 or 1 with probability $\frac{1}{2}$.

Like classical computers with multiple bits, we need multiple qubits to perform any quantum computation. A system with multiple qubits is described by the **tensor product** of the individual vector spaces. Consider a system with two qubits in the $|0\rangle$ state, the combined state is then $|0\rangle \otimes |0\rangle$, usually abbreviated as $|0\rangle^{\otimes 2}$ or $|00\rangle$. The 2-qubit state space is 4-dimensional and has basis $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. Thus, any state vector can be written in terms of this basis,

$$|\psi\rangle = a_{00}\,|00\rangle + a_{01}\,|01\rangle + a_{10}\,|10\rangle + a_{11}\,|11\rangle \tag{2}$$

We can extend this to n qubits. A system with n qubits is a 2^n-dimensional space with basis $\{|x\rangle : x \in \{0,1\}^n\}$, and any state vector can be written as

$$|\psi\rangle = \sum_{x \in \{0,1\}^n} a_x\,|x\rangle \tag{3}$$

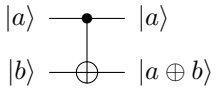
For certain values of a_x, we cannot write $|\psi\rangle$ as tensor product of n states. This gives rise to the amazing phenomenon of **quantum entanglement**. See Appendix A for more discussions.

1.3 Quantum Gates

Next, we will discuss the operations we can perform on qubits, which we call **quantum gates**. We have discussed that the transformations of quantum states are unitary transformations, and in the case of a single qubit, a unitary transformation can be written as a 2×2 unitary matrix. It turns out that it is possible to perform any unitary transformation on a qubit. There are several important quantum gates that are useful in quantum computation. For example, the NOT gate X, the Z gate, and the Hadamard gate,

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{4}$$

There is also an important 2-qubit gate: the controlled-NOT or CNOT gate: $\text{CNOT}(|a\rangle\,|b\rangle) = |a\rangle\,|a \oplus b\rangle$. The CNOT gate flips the second qubit if and only if the first qubit is 1. This gate is usually drawn as follows:



We know that in classical boolean circuits, the AND, OR, and NOT gates are a universal set of gates. Interestingly, there is also a universal set of gates for quantum computation. It is proved that any unitary transformation on multiple qubits can be simulated using CNOT and single qubit gates (e.g. Hadamard, phase, and $\pi/8$ gates).

2 Quantum Algorithms

What are quantum algorithms? In general, an algorithm is a function f that takes an input $x \in \{0,1\}^n$ and calculates $f(x)$. To implement the algorithm f on a quantum computer, we first represent the input x as n qubits, $|x\rangle = |x_1\rangle\,|x_2\rangle \ldots |x_n\rangle$, usually along with some extra qubits in the $|0\rangle$ state. Then, we apply quantum gates (unitary transformations) on the qubits. Finally, we make measurements to obtain an outcome. With clever designs, the outcome of the measurement will be $f(x)$.

In this section, we will describe quantum algorithms that are able to solve problems faster than existing classical algorithms. These algorithms are perhaps the main reasons that people got so excited about quantum computing.

2.1 Quantum Fourier Transform

Mathematically, Fourier transform can be seen as a **change of basis** transformation (see Appendix B for details). We will cover two operations that are crucial in most quantum algorithms, and both can be viewed as Fourier transforms over certain Fourier bases. We will see in the next sections that these two operations are essential in various quantum algorithms.

2.1.1 Quantum Fourier Transform over $\mathbb{Z}_2^n$

A very useful operation is applying n Hadamard gates on n qubits (denoted $H^{\otimes n}$). The state $|x\rangle = |x_1 \dots x_n\rangle$ will be transformed to

$$\begin{aligned} H^{\otimes n} |x_1 \dots x_n\rangle &= \Big(\frac{|0\rangle + (-1)^{x_1} |1\rangle}{\sqrt{2}}\Big)\Big(\frac{|0\rangle + (-1)^{x_2} |1\rangle}{\sqrt{2}}\Big) \cdots \Big(\frac{|0\rangle + (-1)^{x_n} |1\rangle}{\sqrt{2}}\Big) \\ &= \frac{1}{2^{n/2}} \sum_y (-1)^{\sum_i x_i y_i} |y_1 \dots y_n\rangle \\ &= \frac{1}{2^{n/2}} \sum_y (-1)^{x \cdot y} |y\rangle \end{aligned} \tag{5}$$

where $x, y \in \{0,1\}^n$ and $x \cdot y$ is the dot product of x and y. This can be seen as a Fourier transform over Fourier basis

$$\chi_y(x) = \frac{1}{2^{n/2}} (-1)^{x \cdot y} \tag{6}$$

We will see that this operation is used in most quantum algorithms.

2.1.2 Quantum Fourier Transform over $\mathbb{Z}_N$

We will now see the discrete Fourier transform that we are more familiar with,

$$|x\rangle \longrightarrow F |x\rangle = \frac{1}{\sqrt{N}} \sum_y e^{2\pi i x y / N} |y\rangle \tag{7}$$

This is a Fourier transform over basis

$$\chi_y(x) = \frac{1}{\sqrt{N}} e^{2\pi i x y / N} \tag{8}$$

It turns out that we can implement Equation 7 efficiently, with $O(n^2)$ quantum gates. The quantum circuit and the analysis are shown in Appendix C.

Unfortunately, this operation does not mean that we can perform an actual Fourier transform. The reason is that the result of Fourier transform is stored as coefficients of states $|y\rangle$, and any measurement will collapse the state. However, with clever design, this operation can allow us to compute some interesting problems such as factoring!

2.2 Deutsch-Josza Algorithm

The Deutsch-Josza algorithm is a simple yet illustrative example of utilizing the quantum Fourier transform over $\mathbb{Z}_2^n$, which is the $H^{\otimes n}$ operation (Equation 5). Most importantly, we will see that this algorithm provides *exponential* improvement over any possible deterministic classical algorithm.

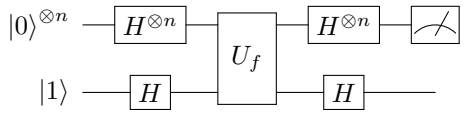
In this problem, there is a function $f : \{0,1\}^n \mapsto \{0,1\}$ with the promise that either

1. $f(x) = 0$ constant for all x, or
2. $f(x)$ is balanced, i.e. equal to 1 for half the possible x.

The goal is to determine which is true: whether f is constant or balanced. In this problem, we can "query" f with values of x, and the performance of the algorithm is measured by the number of queries required.

Classical case. In the worst case, any deterministic algorithm requires $2^{n-1} + 1$ queries.

Quantum case. In order to query f, we assume that there is a unitary transformation U_f acting on $n+1$ qubits such that $U_f |x\rangle |y\rangle = |x\rangle |y \oplus f(x)\rangle$. We will see that we can solve this problem with just a single query of U_f, with the following circuit,



Let's first look at the state $|\psi\rangle$ after applying $H^{\otimes n+1}$ and U_f,

$$\begin{aligned} |\psi_1\rangle = U_f \cdot (H^{\otimes n} |0\rangle^{\otimes n} \otimes H |1\rangle) &= \frac{1}{2^{n/2}} \sum_x |x\rangle \frac{|f(x)\rangle - |f(x) \oplus 1\rangle}{\sqrt{2}} \\ &= \frac{1}{2^{n/2}} \Big(\sum_x (-1)^{f(x)} |x\rangle \Big) \frac{|0\rangle - |1\rangle}{\sqrt{2}} \end{aligned} \tag{9}$$

Interestingly, observe that due to the Hadamard gates, we somehow evaluated $f(x)$ for all x with a single U_f gate, and the result is represented as a sign $(-1)^{f(x)}$. Next, after applying the second Hadamard gates, the state becomes

$$|\psi_2\rangle = \Big(\frac{1}{2^n} \sum_x \sum_z (-1)^{f(x)} (-1)^{z \cdot x} |z\rangle \Big) |1\rangle \tag{10}$$

Finally, we measure the first n qubits. Consider the two cases of f: constant and balanced. We look at the probability of measuring 0^n, i.e. the coefficient of $|0\rangle$.

1. $f(x) = 0$ for all x: the coefficient of $|0\rangle$ is $\frac{1}{2^n} \sum_x (-1)^0 = 1$.
2. f is balanced: the coefficient of $|0\rangle$ is $\frac{1}{2^n} \sum_x (-1)^{f(x)} = 0$.

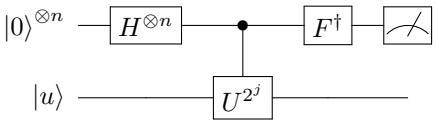
Thus, if f is constant, we will obtain outcome 0^n with probability 1, and if f is balanced, we will obtain outcome 0^n with probability 0. With a single use of U_f, we have deterministically solved the Deutsch-Josza problem!

2.3 Phase Estimation

Now, we will look at an application of the quantum Fourier transform over $\mathbb{Z}_N$. The phase estimation is a key component in other algorithms, including order-finding and factoring (Section 2.4, 2.5).

In this problem, suppose we are given a unitary operator U and an eigenvector $|u\rangle$ with eigenvalue $e^{2\pi i \varphi}$, where $\varphi < 1$ and is unknown. Moreover, we assume that we have access to controlled-U^{2^j} operators for $j = 0, \ldots, n-1$ (later we will see that this is achievable). Our goal is to estimate φ, up to some error $O(2^{-n})$.

Since φ can be any real value, let $\varphi \approx \sigma / 2^n$, where σ is an n-bit integer such that $|\varphi - \sigma/2^n| < 2^{-n}$. We will estimate σ by the following quantum circuit,



where the Controlled-U^{2^j} gate is an abbreviation for (Controlled-$U^{2^{n-1}}$)(Controlled-$U^{2^{n-2}}$) $\cdots$ (Controlled-U^{2^0}), controlled on qubit $|x_1\rangle, \ldots, |x_n\rangle$. The state after Controlled-U^{2^j} is

$$\begin{aligned} |\psi_1\rangle &= \text{Controlled-}U^{2^j} \Big(\frac{1}{2^{n/2}} \sum_x |x\rangle |u\rangle \Big) \\ &= \frac{1}{2^{n/2}} \sum_x e^{2\pi i \varphi (x_1 2^{n-1} + x_2 2^{n-2} + \cdots + x_n 2^0)} |x\rangle |u\rangle \\ &\approx \frac{1}{2^{n/2}} \sum_x e^{2\pi i \sigma x / 2^n} |x\rangle |u\rangle \end{aligned} \tag{11}$$

From Equation 7, the first n qubits are exactly in state $F|\sigma\rangle$. Thus, an inverse Fourier transform $F^\dagger$ will make it $|\sigma\rangle$. Finally, a measurement will give us σ.

Therefore, we can use quantum circuits to efficiently estimate φ up to a small error, where $e^{2\pi i\varphi}$ is the eigenvalue of an operator U. We will see next how this is useful.

2.4 Order-Finding

Given positive integers x and N that are coprime, find the least positive integer r such that $x^r \equiv 1 \pmod N$. Let $n = \lceil \log N \rceil$ bits to represent x and N. Currently, no classical algorithm is known to solve it in $O(poly(n))$ time. We will use phase estimation to *efficiently* solve the order-finding problem. Consider the unitary operator

$$U|y\rangle = |xy \pmod N\rangle \tag{12}$$

We will see that the eigenvalues of U are $e^{2\pi i\varphi}$, and thus phase estimation can be applied. Consider the following state for $s = 0, \ldots, r-1$,

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi isk/r} \left|x^k \pmod N\right\rangle \tag{13}$$

$$U|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi isk/r} \left|x^{k+1} \pmod N\right\rangle = e^{2\pi is/r}|u_s\rangle \tag{14}$$

Thus, $|u_s\rangle$ are eigenstates of U, and $e^{2\pi is/r}$ are the eigenvalues. This is similar to the phase estimation problem. With some clever designs, we can construct the Controlled-U^{2^j} gates, and apply the phase estimation algorithm to compute r.

The total number of gates required is $O(n^3) = O((\log N)^3)$: the construction of the Controlled-U^{2^j} gates requires $O(n^3)$, and the inverse Fourier transform in phase estimation requires $O(n^2)$.

2.5 Shor's Algorithm: Factoring

Finally, we have the tools to perform integer factoring. We will show that we can reduce factoring to order-finding.

Suppose N is an odd composite integer. To factor N, we need two theorems from number theory:

1. For an integer x, if $x^2 = 1 \pmod N$ and $x \neq \pm 1 \pmod N$, then at least one of $gcd(x-1, N)$ and $gcd(x+1, N)$ is a factor of N.
2. For a uniformly chosen y such that $1 \leq y \leq N-1$ and y is a coprime to N, with high probability, the order r of y is even and satisfies $y^{r/2} \neq -1 \pmod N$.

The second theorem shows that we will likely pick a y such that $y^r = 1 \pmod N$ and $y^{r/2} \neq \pm 1 \pmod N$. Then, we use the order-finding algorithm to obtain r, and compute $x = y^{r/2}$. Finally, the first theorem shows that we will obtain a factor which is $gcd(x-1, N)$ or $gcd(x+1, N)$.

Let N be represented as $n = \lceil \log N \rceil$ bits. The procedure is as follows:

1. If N is even, output 2.
2. Determine whether $N = a^b$ for some integers a, b, with an $O(n^3)$ classical algorithm.
3. Randomly select y from 1 to $N-1$. If $gcd(y, N) > 1$, then output $gcd(y, N)$.
4. Use order-finding to obtain the order r of y, with an $O(n^3)$-size quantum circuit.
5. If r is even and $y^{r/2} \neq -1 \pmod N$, then compute $x = y^{r/2}$, $gcd(x-1, N)$ and $gcd(x+1, N)$, and output one that is a non-trivial factor. Otherwise, go back to step 3.

This shows that we can perform factoring in $O(poly(n))$ time. In contrast, the best known classical algorithm takes superpolynomial time.

3 Quantum Complexity

This section focuses on the question of "*What problems can quantum computers solve?*" In this section, we define the class BQP, and look at some known results and open problems for this class. Please refer to [4] for more advanced topics.

3.1 BQP

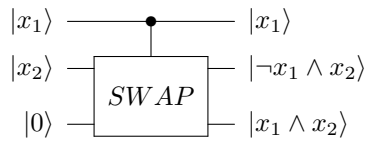
Definition 3.1 *BQP is the class of languages L for which there exists a quantum circuit C and a polynomial q such that for all n and input $x \in \{0,1\}^n$,*

- *If $x \in L$, then Pr$[C(|x\rangle |0\rangle^{\otimes q(n)}) = 1] \geq \frac{2}{3}$.*
- *If $x \notin L$, then Pr$[C(|x\rangle |0\rangle^{\otimes q(n)}) = 1] \leq \frac{1}{3}$.*

The circuit can be defined more rigorously as a uniform family of quantum circuits $\{C_n\}$. The input is encoded as a state $|x\rangle$, and $|0\rangle^{\otimes q(n)}$ are the extra qubits required for the computation. This definition is very similar to the definition of BPP. Next, we will see where BQP sits within the classical complexity classes.

Theorem 3.1 *$P \subseteq BQP$.*

It seems trivial that a quantum computer can simulate a classical computer. However, we still need to address some fundamental differences between classical and quantum circuits. Recall that classical circuits consist of AND, OR, and NOT gates, and quantum circuits consist of unitary operators. The fundamental difference is that the classical AND and OR gates are *not* reversible, whereas all unitary operators are reversible ($U^\dagger U = I$). Nevertheless, we will show that we can simulate these irreversible gates by adding an extra qubit. For example, $x_1 \wedge x_2$ can be computed by



Thus, for any polynomial-size classical circuit, we can simulate it with a polynomial-size quantum circuit, along with polynomial number of extra qubits.

Theorem 3.2 *$BPP \subseteq BQP$.*

Given that quantum circuits can simulate classical circuits, we just need some random bits. We can obtain a true random bit by making a measurement on $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, giving 0 and 1 with probability $1/2$ each.

Theorem 3.3 *$BQP \subseteq EXP$.*

Any quantum state with n qubits is written as $|\psi\rangle = \sum_{x\in\{0,1\}^n} a_x |x\rangle$, a superposition of at most 2^n orthogonal states. A classical algorithm can simulate the transformations of a quantum algorithm $U|\psi\rangle = \sum_x a_x U|x\rangle$ in exponential time.

Theorem 3.4 *$BQP \subseteq PSPACE$.*

Suppose a quantum algorithm takes T steps, and the quantum circuit consists of 1-, 2-, or 3-qubit gates, $U_1, U_2, \ldots, U_T$. Let the final state be $\left|\psi^{(T)}\right\rangle = U_T U_{T-1} \cdots U_1 \left|\psi^{(0)}\right\rangle$. To simulate a quantum algorithm and obtain the final result, we need to calculate the probability of the final state being in certain states $|y\rangle$, i.e. calculate $|\langle y|\psi^{(T)}\rangle|^2$. To do so in polynomial space, we can use the trick that $\sum_x |x\rangle\langle x| = I$, and rewrite the inner product as

$$\langle y|\psi^{(T)}\rangle = \sum_{x_1 x_2 \ldots x_T} \langle y|U_T|x_{T-1}\rangle \langle x_{T-1}|U_{T-1}|x_{T-2}\rangle \cdots \langle x_1|U_1|\psi^{(0)}\rangle \tag{15}$$

Evaluating each term in the summation requires only polynomial space, and we can reuse space. Thus, this can be done in polynomial space.

Another way to show this is by a recursive algorithm. Let $\left|\psi^{(i)}\right\rangle = \sum_x a_x^{(i)} |x\rangle$ be the state at time step i. Our goal is to obtain $a_x^{(T)}$ recursively. Since the transformation from step $i-1$ to i is on at most 3 qubits, $a_x^{(i)}$ can be determined from 8 values of $a_{y_j}^{(i-1)}$, where y_j and x only differ in the 3 qubits being operated. The space for calculating $a_{y_j}^{(i-1)}$ can be reused. Thus, for each x and i, the space for calculating $a_x^{(i)}$ is $S(i) \leq S(i-1) + O(1)$. This procedure allows us to determine the final coefficients $a_x^{(T)}$ in polynomial space, i.e. quantum algorithms can be computed in polynomial space.

Theorem 3.5 *BQP $\subseteq$ PP.*

PP is the class of problems solved by probabilistic Turing machines with error probability $< 1/2$. In order to show BQP $\subseteq$ PP, we first think of quantum algorithms as a tree of paths. For example, $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ can be seen as choosing a path to $|0\rangle$ or $|1\rangle$, each with probability $1/2$. The final result is obtained by looking at the final state of a path. For quantum circuits, there are more correct paths (paths leading to the correct answer) than wrong paths. Thus, we can build a PP algorithm based on randomly selecting two paths and looking at their final states.

3.2 Open Problems

Just like classical complexity theory, there are also many open problems in quantum complexity. So far, we have the following:

$$\text{P} \subseteq \text{BPP} \subseteq \text{BQP} \subseteq \text{PP} \subseteq \text{PSPACE} \subseteq \text{EXP} \tag{16}$$

We know that P $\neq$ EXP, so one of the inclusions must be strict. In fact, people believe that all inclusions are strict (except the first one). However, we haven't even proved that P $\neq$ PSPACE. Separating BPP from BQP (proving BPP $\neq$ BQP) would imply that P $\neq$ PSPACE.

We do not know the relation between BQP and NP. However, we can gain some insights from Grover's algorithm for unstructured search: given a boolean function f, find an element $x \in \{x_1, \ldots, x_N\}$ such that $f(x) = 1$. Grover's algorithm can achieve this using $O(\sqrt{N} \log N)$ gates. This means that we have a quantum algorithm to solve SAT in $O(n2^{n/2})$ time. This is a quadratic speedup over classical algorithms. However, this is far from proving that NP $\subseteq$ BQP.

4 Quantum Merlin-Arthur

Recall that a language L in NP is such that for every $x \in L$, there is a certificate that a Turing machine can verify. We can try to make a quantum version of NP. Since quantum algorithms are usually probabilistic, it is natural to make analogue to the complexity class MA.

Definition 4.1 *QMA is the class of languages L for which there exists a quantum circuit C such that for all inputs x,*

- *If $x \in L$, then there exists a state $|\varphi\rangle$ such that Pr$[C(|x\rangle, |\varphi\rangle) = 1] \geq \frac{2}{3}$.*
- *If $x \notin L$, then for any state $|\varphi\rangle$, Pr$[C(|x\rangle, |\varphi\rangle) = 1] \leq \frac{1}{3}$.*

Here $|\varphi\rangle$ is the certificate that Merlin sends to Arthur. It is important to note that the certificate is a quantum state, not a string. This raises two concerns that we do not have in the classical case. First, in order to perform success amplification, we would like to run the verifier several times. However, due to the no-cloning theorem, it is impossible to replicate the certificate $|\varphi\rangle$. Secondly, the devious Merlin can send Arthur a state that is entangled with other states, which to Arthur is a mixed state. It turns out that we do not need to worry about these two concerns: Merlin can achieve the highest error probability by sending duplicates of a pure state $|\varphi\rangle^{\otimes T}$.

We can define another complexity class if the certificate is classical but the verifier is a quantum circuit. We call this class Quantum Classical MA (QCMA).

4.1 Complexity of QMA

Theorem 4.1 *MA $\subseteq$ QCMA $\subseteq$ QMA.*

This result is quite straightforward. The first inclusion follows from the fact that quantum circuits can simulate classical circuits. The second inclusion follows from that we can encode the proof from Merlin in a quantum state, and simulate the verifier with a quantum circuit.

Theorem 4.2 *QMA $\subseteq$ PP.*

The proof is similar to the proof for BQP $\subseteq$ PP.

4.2 QMA Example: The Local Hamiltonian Problem

The k-Local Hamiltonian problem (k-LH) is an example of a problem in QMA. This problem has some physics motivation: a Hamiltonian is an operator corresponding to the energy of a physical system, and the smallest eigenvalue is the energy of the ground state. In addition, we will show in Section 4.3 that this problem is QMA-complete.

Definition 4.2 *k-Local Hamiltonians problem (k-LH problem).*

- ***Input**: $\{H_1, \ldots, H_r, a, b\}$. $H_1, \ldots H_r$ are Hermitian positive semi-definite matrices that operate on only k qubits, with $\|H_i\| \leq 1$. a and b are real numbers such that $b - a > 1/poly(n)$.*
- ***Output**: Let λ_0 be the smallest eigenvalue of $H = H_1 + \cdots H_r$. Output 1 if $\lambda_0 \leq a$, and output 0 if $\lambda_0 \geq b$.*

Theorem 4.3 *k-LH is in QMA for any $k = O(\log n)$ and $b - a > 1/poly(n)$.*

If $\lambda_0 \leq a$, then Merlin can simply send the corresponding eigenstate $|\varphi_0\rangle$. The verifier V first randomly selects an H_i, and performs a measurement based on POVM $\{E_0, E_1\}$ where $E_0 = H_i$ and $E_1 = I - H_i$ (this is a more generic notion of a measurement). The outcome will be 1 with probability $\langle\varphi_0|I - E_1|\varphi_0\rangle = 1 - \langle\varphi_0|H_i|\varphi_0\rangle$. Thus, in total the probability of outputting 1 is

$$\Pr[V(H, |\psi_0\rangle) = 1] = \frac{1}{r}\sum_{i=1}^{r}(1 - \langle\varphi_0|H_i|\varphi_0\rangle) = 1 - \frac{\lambda_0}{r} \geq 1 - \frac{a}{r} \tag{17}$$

On the other hand, if $\lambda_0 \geq b$, then for any state $|\varphi\rangle$ that Merlin sends, $\langle\varphi|H|\varphi\rangle \geq \langle\varphi_0|H|\varphi_0\rangle) = \lambda_0$. The probability that the measurement will output 1 is

$$\Pr[V(H, |\psi\rangle) = 1] = 1 - \frac{1}{r}\langle\varphi|H|\varphi\rangle \leq 1 - \frac{\lambda_0}{r} \leq 1 - \frac{b}{r} \tag{18}$$

Thus, if $b - a > 1/poly(n)$, we can amplify this probability gap. This shows that k-LH is in QMA.

4.3 QMA-completeness

It turns out that k-Local Hamiltonian is not only in QMA, but QMA-complete! The full proof is complicated and will not be shown here. Interestingly, QMA-completeness is first proved for $k \geq 5$ [1], and a few years later for $k \geq 3$ and $k \geq 2$ [2].

We will first show that k-LH is closely related to k-SAT, and since SAT is NP-complete, it is somewhat intuitive that k-LH is QMA-complete. Then, we will give a high-level explanation of the reduction from QMA problems to k-LH, which has a similar procedure to Cook-Levin Theorem.

4.3.1 Relationship with k-SAT

We first show that k-SAT can be reduced to the k-LH problem. Let a k-CNF formula be $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_r$ with n variables, where each C_i is a disjunctive clause of k literals. For a clause C_i, let x_i be the unique unsatisfying assignment, i.e. $C_i(x_i) = 0$, and define $H_i = |x_i\rangle\langle x_i| \otimes I_{others}$,

the projection matrix that projects the corresponding k qubits onto $|x_i\rangle$. We can see that for n qubits in state $|z\rangle$, if z satisfies C_i, then $H_i |z\rangle = 0$, and if z does not satisfy C_i, then $H_i |z\rangle = |z\rangle$.

Thus, for an assignment z, $H |z\rangle = (H_1 + \cdots + H_r) |z\rangle = m |z\rangle$, where m is the number of clauses not satisfied by z. If ϕ is satisfiable, then there exists a z such that $H |z\rangle = 0$, and since H is positive semi-definite, its smallest eigenvalue is 0. On the other hand, if ϕ is unsatisfiable, then for all z there must be at least one C_i that's not satisfied, i.e. $m \geq 1$. In fact, the smallest eigenvalue $m_{\min}$ is equal to r - (the maximum number of satisfiable clauses), which corresponds to the MAX-SAT problem.

4.3.2 Local Hamiltonian is QMA-complete

Theorem 4.4 *For $k \geq 2$, k-Local Hamiltonian is a QMA-complete problem.*

We will only give a high-level explanation of reducing QMA problems to local Hamiltonian. Rigorous proofs and detailed analysis can be found in [1, 2]. Recall the idea of Cook-Levin theorem for NP problems: using local checks to verify that a computation history is valid and has output 1. We will try to mimic this procedure. Suppose the quantum verifier V operating on input $|x\rangle$ and certificate $|\varphi\rangle$ consists of T operations, $U_T U_{T-1} \cdots U_2 U_1 |x\rangle |\varphi\rangle$. Then, we use local Hamiltonians to verify the time evolution of the quantum states. How do we verify that at each time step t, the transformation from $t-1$ to t is correct?

Consider the following state, which is a superposition of the tensor product of the state at time t and an extra "clock" state $|t\rangle$, over all time steps.

$$|\eta\rangle = \frac{1}{\sqrt{T+1}} \sum_{t=0}^{T} U_t U_{t-1} \ldots U_1 |x\rangle |\varphi\rangle \otimes |t\rangle \tag{19}$$

And consider the following Hermitian operator,

$$H_t = \frac{1}{2}\big(I \otimes |t\rangle\langle t| + I \otimes |t-1\rangle\langle t-1| - U_t \otimes |t\rangle\langle t-1| - U_t^\dagger \otimes |t-1\rangle\langle t|\big) \tag{20}$$

It is easy to see that $H_t |\eta\rangle = 0$. This verifies that the transformation from time $t-1$ to t is indeed the operation U_t. Along with operators H_{in} checking the input $|x\rangle$ and H_{out} checking the final state, we have obtained a set of Hamiltonians, and the sum of the Hamiltonians H satisfies $H |\eta\rangle = 0$. Note that we have not set constraints on $|\varphi\rangle$. Thus, if there exists a certificate $|\varphi\rangle$ such that $V(|x\rangle, |\varphi\rangle)$ accepts with high probability, then the smallest eigenvalue of H will be close to 0.

A lot more work must be done to show that the checks can be done using 2-local Hamiltonians, along with detailed analysis of the error probability. Nevertheless, the above shows the high-level idea of reducing a problem in QMA to the Local Hamiltonian problem.

5 Conclusion

In this paper, we have shown several important results of quantum computation and quantum complexity theory. We have shown quantum algorithms that provide exponential speedup over existing classical algorithms, and we have discussed the BQP and QMA complexity classes. As we saw, there are still many open questions in quantum complexity theory, and quantum computation remains an exciting and active research field to physicists and computer scientists.

References

[1] D. Aharonov and T. Naveh. Quantum np-a survey. *arXiv preprint quant-ph/0210077*, 2002.

[2] J. Kempe, A. Kitaev, and O. Regev. The complexity of the local hamiltonian problem. *SIAM Journal on Computing*, 35(5):1070–1097, 2006.

[3] M. A. Nielsen and I. Chuang. Quantum computation and quantum information, 2002.

[4] R. O'Donnell. Quantum computation and information 2015. *https://www.cs.cmu.edu/ odonnell/quantum15/*, 2015.

Appendix

A Quantum Entanglement

Quantum entanglement is a weird yet amazing phenomenon. Consider the following 2-qubit states,

$$\begin{aligned}
|\psi_{00}\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\
|\psi_{01}\rangle &= \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \\
|\psi_{10}\rangle &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \\
|\psi_{11}\rangle &= \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)
\end{aligned} \tag{21}$$

These are 4 orthogonal states known as the *Bell states*. For any of these, we cannot write it as a product of two 1-qubit states, which means that it is entangled.

These states can be obtained from a Hadamard and a CNOT gate. For example,

$$\text{CNOT}(H|0\rangle \otimes |0\rangle) = \text{CNOT}(\frac{|0\rangle + |1\rangle}{\sqrt{2}}|0\rangle) = \frac{|00\rangle + |11\rangle}{\sqrt{2}} = |\psi_{00}\rangle \tag{22}$$

Why is entanglement so interesting? We will show a surprising application: *superdense coding*. Suppose Alice and Bob shares a pair of qubits in the state $|\psi_{00}\rangle$, and Alice wants to send Bob a 2-bit message $x \in \{0,1\}^2$. Alice can perform operations on her own qubit such that the state becomes one of the Bell states $|\psi_x\rangle$, and then she sends her qubit to Bob. Now, since the Bell states are orthogonal, Bob can easily measure the state and determine x.

This is interesting because Alice transmits 2 bits of information to Bob by sending only 1 qubit (1 use of the quantum channel). This cannot be done classically. Superdense coding is just one of many interesting applications of quantum entanglement.

B Generic Notion of Fourier Transform

Consider a function $f : G \to \mathbb{C}$. f can be seen as a vector in a vector space of dimension $|G|$, in the standard basis $\{\delta_k\}$. Suppose $\{\chi_k\}$ is an orthonormal basis in the vector space, then

$$f(x) = \sum_{y \in G} f(y)\delta_y(x) = \sum_{y \in G} \hat{f}(y)\chi_y(x) \tag{23}$$

where $\hat{f}(y)$ are the coefficients of f in the new basis, given by

$$\hat{f}(y) = \sum_{x \in G} f(x)\chi_y(x) \tag{24}$$

The transformation of f to $\hat{f}$ is simply a change of basis, where we call $\{\chi_k\}$ a Fourier basis. This is the **change of basis view of Fourier transform**.

In the quantum case, we would like to design quantum algorithms that transform a basis state to

$$|x\rangle \longrightarrow \sum_y \chi_y(x)|y\rangle \tag{25}$$

From Equation 24, the algorithm will transform any state $|\psi\rangle = \sum_x f(x)|x\rangle$ to

$$|\psi\rangle = \sum_x f(x)|x\rangle \longrightarrow \sum_y \hat{f}(y)|y\rangle \tag{26}$$

This is essentially transforming the coefficients f to $\hat{f}$.

C Quantum Fourier Transform over $\mathbb{Z}_N$

We would like implement the following transformation (Equation 7) with quantum circuits.

$$|x\rangle \longrightarrow F|x\rangle = \frac{1}{\sqrt{N}} \sum_y e^{2\pi i xy/N} |y\rangle \tag{27}$$

Suppose $N = 2^n$, so x, y can be expressed as $x = \sum_{k=1}^n x_k 2^{n-k}$ and $y = \sum_{k=1}^n y_k 2^{n-k}$. Let $\omega = e^{2\pi i/2^n}$. We first rewrite Equation 27 as a tensor product of n states,

$$\begin{aligned} F|x\rangle &= \frac{1}{2^{n/2}} \sum_{y_1 \dots y_n} \omega^{x \cdot \sum_{k=1}^n y_k 2^{n-k}} |y_1 \dots y_n\rangle \\ &= \frac{1}{2^{n/2}} \sum_{y_1 \dots y_n} \bigotimes_{k=1}^n \omega^{x y_k 2^{n-k}} |y_k\rangle \\ &= \frac{1}{2^{n/2}} \bigotimes_{k=1}^n \Big(|0\rangle + \omega^{x \cdot 2^{n-k}} |1\rangle \Big) \\ &\equiv |\psi_1\rangle |\psi_2\rangle \cdots |\psi_n\rangle \end{aligned} \tag{28}$$

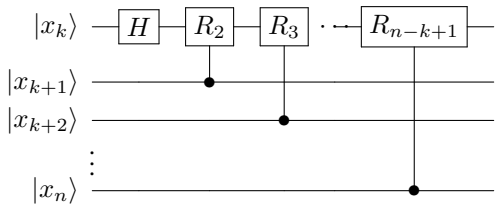
where $|\psi_k\rangle = \frac{1}{\sqrt{2}}(|0\rangle + \omega^{x \cdot 2^{n-k}} |1\rangle)$. Observe that

$$\omega^{x \cdot 2^{n-k}} = e^{2\pi i x 2^{-k}} = e^{2\pi i \sum_{l=1}^n x_l 2^{n-k-l}} = e^{2\pi i (\frac{x_{n-k+1}}{2} + \frac{x_{n-k+2}}{2^2} + \cdots + \frac{x_n}{2^k})} \tag{29}$$

It turns out that we can construct an *efficient* quantum circuit to compute $|\psi_1\rangle \dots |\psi_n\rangle$, using Hadamard operators, the following quantum gates,

$$R_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{bmatrix} \tag{30}$$

and the quantum circuit below (for $|\psi\rangle_{n-k+1}$),



The state $|x_k\rangle$ is transformed to

$$|x_k\rangle \longrightarrow \frac{1}{\sqrt{2}} \Big(|0\rangle + (-1)^{x_k} e^{2\pi i (\frac{x_{k+1}}{2^2} + \cdots + \frac{x_n}{2^{n-k+1}})} \Big) = |\psi_{n-k+1}\rangle \tag{31}$$

This can be seen from Equation 29. Thus, with a combination of these circuits, $|x_1\rangle |x_2\rangle \cdots |x_n\rangle$ is transformed to $|\psi_n\rangle |\psi_{n-1}\rangle \cdots |\psi_1\rangle$. After $n/2$ swap operations, we get our final result, $|\psi_1\rangle |\psi_2\rangle \cdots |\psi_n\rangle$.

The gates used in the circuit are n Hadamard gates, $n(n-1)/2$ Controlled-R^k gates, and $n/2$ swap gates. Therefore, the quantum algorithm can be performed with $O(n^2)$ gates. In contrast, the best classical algorithm (such as FFT) for $N = 2^n$ elements requires $\Theta(n2^n)$ gates.