Quantum Computing

Andrzej Antoni Kwaśniewski

Jagiellonian University
Faculty of Mathematics and Computer Science
Institute of Theoretical Computer Science

Advisor:
Prof. Michał Wrona

Kraków
2025 CE

Contents

Introduction

This thesis serves as a brief introduction to quantum computing. We begin with an overview of quantum mechanics, followed by relevant notation and linear algebra concepts. Later, we show the most influential quantum algorithms and the quantum complexity classes they represent. The content is heavily based on chapter 20 of the book ”Computational Complexity: A Modern Approach” by S. Arora and B. Barak [**?**]. The aim is to explain the subject in an easy-to-follow way without sacrificing formality.

1

Preliminaries

Not all of the contents of this chapter are strictly necessary for further study of quantum computation, its meant so the reader can get acquainted with the intuition behind quantum mechanics and its interpretation to better understand the algorithms.

1.1 Wave function

In classical mechanics, the state of a particle is fully described using its position and momentum. In quantum it is described by wave function - Ψ, which is a solution to the Schrödinger equation

$$i\hbar\frac{d}{dx}\psi = -\frac{\hbar^2}{2m}\nabla^2\psi + V(x,y)\psi$$

In spite of ψ being the solution to the equation, the physically measurable value is $|\psi|^2$. All the mathematical formalism matches the physical experiments. However, we are left with the conundrum of how to interpret this.

1.2 Probabilistic Interpretation

According to the most widely accepted explanation postulated by Niels Bohr - the Copenhagen interpretation - $|\psi|^2$ represents a probabilistic density of the position of the particle in the given moment.

$$|\psi(\vec{x},t)|^2 = \rho(\vec{x},t)$$

Moreover this interpretation stipulates that before we measure the state of the particle it is in no particular state, it is in superposition of all the possible spaces. Only after measurement does its state change immediately at random with $|\psi|^2$ as its density function, which is called quantum wave collapse.

1.3 Notation

Quantum states form vector spaces on $\mathbb{C}$, known as the Hilbert space and denoted as $\mathcal{H}$, their members are column vectors, which in dirac notation are denoted as $|\psi\rangle$.
We can similarly define a dual space $\mathcal{H}^*$ - the space of functions $\mathcal{H} \to \mathbb{C}$, its members are row vectors in the Dirac notation denoted as $\langle\psi|$.

$\langle\psi|$ is a hermitian conjugate of $|\psi\rangle$, which means that if

$$|\psi\rangle = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$$

then

$$\langle\psi| = \begin{bmatrix} a_1^* & a_2^* & \dots & a_n^* \end{bmatrix}$$

We can thus define an inner (scalar) product between $|\psi\rangle = \begin{bmatrix} a_1 & \dots & a_n \end{bmatrix}^T$ and $|\theta\rangle = \begin{bmatrix} b_1 & \dots & b_n \end{bmatrix}^T$ as

$$\langle\psi|\theta\rangle = a_1^* b_1 + a_2^* b_2 + \dots + a_n^* b_n$$

1.4 Measurement

As we discussed earlier, before measurement the quantum state is in no particular state, but only after measurement it collapses to one of the states. We can calculate the probability of obtaining the state $|\phi\rangle$ using inner product

$$P(|\phi_i\rangle) = |\langle\phi_i|\psi\rangle|^2$$

We interpreted ψ as the density function of probability, therefore it must be normalised, $\langle\psi|\psi\rangle = 1$, to ensure that the probabilities of obtaining the states sum up to 1.

1.5 Operators

Operators are functions $\mathcal{H} \to \mathcal{H}$, for operator Q:

$$Q|\psi\rangle = |Q\psi\rangle = |\psi'\rangle \in \mathcal{H}$$

We commonly use shorthand notation

$$\langle\psi|Q|\theta\rangle := \langle\psi|Q\theta\rangle$$

In quantum mechanics we typically only use linear operators, such that

$$Q(a_1|\psi_1\rangle + a_2|\psi_2\rangle) = a_1 Q|\psi_1\rangle + a_2 Q|\psi_2\rangle$$

Hermitian adjoint $Q^\dagger$ to Q is an operator such that

$$\langle\psi|Q\theta\rangle = \langle Q^\dagger\psi|\theta\rangle$$

Operator Q is called hermitian if $Q = Q^\dagger$.

1.6 Qubit

The qubit is the quantum analogue to the bit; it is the simplest quantum mechanical system. It has a two-dimensional state space, and thus its orthonormal basis is formed by two orthogonal vectors $|0\rangle$ and $|1\rangle$. By convention, we use $|0\rangle = [1, 0]$ and $|1\rangle = [0, 1]$. An arbitrary qubit $|\psi\rangle$ can therefore be fully described as

$$|\psi\rangle = a|0\rangle + b|1\rangle$$

Qubit must satisfy the normalization factor for state vectors

$$\langle\psi|\psi\rangle = 1$$

which for qubits means that

$$a^2 + b^2 = 1$$

The key difference between a bit and a qubit is that when a bit can only be in either state 0 or 1, qubit can be in superposition of states $|0\rangle, |1\rangle$, meaning that it is in both states at once and only after measurement it becomes one of them with probabilities proportional to a, b.

1.7 Quantum gates

Quantum gates are quantum operators operating on a reduced number of qubits. Similar to classical logic gates, they are the building block of quantum circuits. The most commonly used gates are

- Pauli-X, the not gate $X = \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
- Pauli-Y, $Y = \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$
- Pauli-Z, $Z = \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
- Hadamard, $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$
- Phase shift - $P(\phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}$, it maps $|0\rangle \to |0\rangle$ and $|1\rangle \to e^{i\phi}|1\rangle$.

1.8 Entanglement

Consider the following two qubit state

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{1}$$

this notation means that either both qubits are $|0\rangle$ or both are $|1\rangle$. If there are no single qubits $|a\rangle$, $|b\rangle$ such that $|\psi\rangle = |a\rangle + |b\rangle$ then we say that $|\psi\rangle$ is an entangled state.

EPR paradox

Quantum entanglement is a fundamental concept from which we can conclude seemingly paradoxical results.

Consider two entangled qubits from the states $|\psi\rangle$, A and B. Before measurement they are in superposition between states $|0\rangle$ and $|1\rangle$ and entangled in such a way that after measurement both become $|0\rangle$ or both become $|1\rangle$. Now, suppose that we give qubit A to Alice and B to Bob, Alice, and Bob then travel far apart. If Alice performs a measurement on her qubit, it instantaneously collapses to one of the states. Bob can then quickly measure his qubit and know exactly what value Alice's qubit is in, in spite of the distance between them. This process seems to convey information faster than light.

This process is a simplification of the famous EPR paradox formed by Einstein Podolsky and Rosen in 1935 [?]. It can have two possible explanations; either during the thought experiment, we made an incorrect assumption or there is some hidden variable in the qubits.
We made the following assumptions.

- (Completeness) Every element of reality has a counterpart in theory
- (Reality) We can predict with certainty value of physical quantity without disturbing the system.
- (Locality) Element cannot be instantaneously affected by measurements performed on another system distant from them

As Bell showed theoretically [?], and then later was shown experimentally by Alain Aspect [?] quantum mechanics cannot be explained by including any hidden variables, therefore if we assume quantum mechanics to be complete either reality or locality assumption must be abandoned.

2

Grover's algorithm

2.1 Problem statement

Grover's algorithm solves unstructured search, given an oracle function and desired output y it computes the value x for which $f(x) = y$.

Formally we are given an oracle function

$$f : \{0,1\}^n \to \{0,1\}$$

And we have to find an input $x \in \{0,1\}^n$ such that $f(x) = 1$.

We cannot assume anything about the function properties. Therefore, without quantum search, the best approach is to guess randomly. For M possible solutions to the problem, the expected number of attempts before finding a solution is $\frac{N}{M}$, where $N = 2^n$. In contrast, Grover's algorithm can solve this task using just $\sqrt{\frac{N}{M}}$ function calls.

2.2 The algorithm

The algorithm requires n qubits in the register, enough to represent all states in the function domain. We start by using n Hadamard gates to set the register to the uniform state - the superposition of all states in the domain, resulting in the state $|\varphi\rangle$:

$$|\varphi\rangle = \frac{1}{M} \sum_x |x\rangle$$

Now we use the fact that all x evaluate either to 0 or 1

$$\forall_x : \left[x \in f^{-1}(0) \lor x \in f^{-1}(1)\right]$$

and split the sum based on that

$$|\varphi\rangle = \frac{1}{\sqrt{N}} \left(\sum_{x \in f^{-1}(0)} |x\rangle + \sum_{x \in f^{-1}(1)} |x\rangle \right)$$

now we define $|a\rangle$ and $|b\rangle$ as a superposition of all possible states in $f^{-1}(0)$ and $f^{-1}(1)$ respectively

$$|a\rangle = \frac{1}{\sqrt{N-M}} \sum_{x \in f^{-1}(0)} |x\rangle \qquad |b\rangle = \frac{1}{\sqrt{M}} \sum_{x \in f^{-1}(1)} |x\rangle$$

we use $|a\rangle$ and b to yet again rewrite our state

$$|\varphi\rangle = \sqrt{\frac{N-M}{N}}\, |a\rangle + \sqrt{\frac{M}{N}}\, |b\rangle$$

we now use the fact $\sqrt{\frac{N-M}{N}}^2 + \sqrt{\frac{M}{N}}^2 = 1$ and $\sqrt{\frac{N-M}{N}} \leq 1$ and $\sqrt{\frac{M}{N}} \leq 1$ to parametrize

$$\cos\theta = \frac{N-M}{N} \qquad \sin\theta = \frac{M}{N}$$

$$|\phi\rangle = \cos\theta\,|a\rangle + \sin\theta\,|b\rangle$$

We now define a quantum gate Q_A that will flip the phase of the state if $x \in f^{-1}(1)$

$$Q_A\,|x\rangle = (-1)^f(x)\,|x\rangle$$

see that we can rewrite Q_A as

$$Q_A\,|x\rangle = |a\rangle\,\langle a|\,|x\rangle - |x\rangle$$

we can interpret this as a reflection through the $|a\rangle$-axis, this can be easily verified, using orthogonality of $|a\rangle$ and $|b\rangle$

$$Q_A\,|a\rangle = |a\rangle \qquad O\,|b\rangle = -\,|b\rangle$$

Analogously, we define an operator of reflection through $|\phi\rangle$ axis

$$Q_S\,|x\rangle = 2\,|\phi\rangle\,\langle\phi|\,|x\rangle - |x\rangle$$

if we interpret those rotations geometrically, and assume that $M \ll N$, in the beginning we had an angle of $|\theta\rangle$ between $|a\rangle$ and $|\phi\rangle$. After we act on $|\phi\rangle$ with $Q_S \cdot Q_A$ we get a new state $|\phi'\rangle$, which is a rotation of $|\phi\rangle$ by 2θ towards $|b\rangle$. We attach an illustration of this process, when plotted on the Bloch sphere 1.

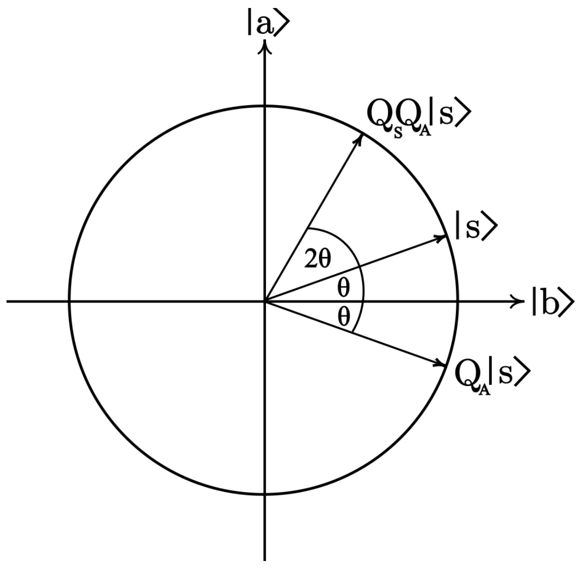

Figure 1: One step of Grover's search

2.3 Complexity

In each step we change the angle of our state by 2θ towards $|b\rangle$ (note that θ is the same in each iteration). If we approximate θ as $\sqrt{\frac{N}{M}}$ we find that we need roughly $\frac{\pi}{4}\sqrt{\frac{N}{M}}$ steps to rotate all the way to $|b\rangle$. Then, if we measure a state close to $|b\rangle$, the probability of obtaining a valid answer is almost certain. Overall, algorithm complexity is $O(\sqrt{n})$.

2.4 Impact

Grover's algorithm solves unstructured search, an `NP`-complete problem with $O(\sqrt{N}) = O(\sqrt{2^N})$ steps; furthermore, we know that the algorithm is optimal - on quantum computers, there is no algorithm solving this problem with lower complexity [**?**]. This means that using a quantum computer, we only get a quadratic speed-up on the problem, which makes it remain in `NP`. This result led most of the experts to believe that `NP` complete problems are not possible to solve efficiently on a quantum computer.

3

Simon's Algorithm

Simon's algorithm was one of the first to show quantum advantage; it solves an NP-hard problem in polynomial time using a quantum computer.

Given a polynomial-time computable function $f : \{0,1\}^n \to \{0,1\}^n$ it finds value $s \neq 0^n$ for which $\forall x : f(x) = f(x \otimes s)$ if such s exists. The function f is an oracle, meaning that we don't have direct knowledge of how it computes its values.

3.1 The algorithm

In the algorithm we perform a quantum subroutine n times. The subroutine uses $2n$ qubits initialized to $|0\rangle$. Qubits are split into two registers with length n. We first use n Hadamard gates to set the first register to a uniform state, resulting in the state

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle |0\rangle$$

then we apply function f to the second n bits giving us

$$\frac{1}{\sqrt{2^n}} \sum_x |x\rangle |f(x)\rangle$$

and then again apply the Hadamard gate to all qubits in the first register

$$\frac{1}{2^n} \sum_{x,y} (-1)^{xy} |y\rangle |f(x\rangle$$

if we change the sum to sum over all images or f rather than inputs we can write as

$$\begin{aligned} &\frac{1}{2^n} \sum_{y,f(a)} ((-1)^{a\cdot y} + (-1)^{(a\oplus s)\cdot y} |y\rangle |f(a)\rangle = \\ &= \frac{1}{2^n} \sum_{y,f(a)} (-1)^{a\cdot y}(1 + (-1)^{sy}) |y\rangle |f(a)\rangle \end{aligned}$$

if $f(x)$ was in the second register before the last gate, then in the first register there is either a or $a \oplus s$ as $f(a) = f(a \oplus s)$.

Now, we notice that if two solutions $(a, a \oplus s)$ exist then it must be that $s \cdot y = 0$. We are only summing over values where two solutions exist thus we get

$$\frac{1}{2^{n-1}} \sum_{y,f(a)} (-1)^{a\cdot y} |y\rangle |f(a)\rangle$$

if we then perform a measurement on the first n qubits, y is selected among all values such that

$$s \cdot y = 0$$

By itself y is not enough to compute s, we need to repeat this subroutine until we find $n-1$ linearly independent equations $(s \cdot y_1, \ldots, s \cdot y_n)$. We only need $n-1$ rather than n values, as we already assumed that $s \neq 0^n$.

Once we have $n-1$ linearly independent solutions, we can find s classically in polynomial time using Gaussian elimination [**?**].

We still have to bound the probability that resulting equations are linearly independent. Equation s_i is linearly independent if it can't be expressed as a combination of rows $\{1, s_{i-1}\}$, therefore the probability of all equations being linearly independent is

$$\prod_{k=0}^{n-1}\left(1-\frac{2^k}{2^n}\right)=\prod_{k=0}^{n-1}\left(1-\frac{1}{2^{n-k}}\right)=\prod_{k=1}^{n-1}\left(1-\frac{1}{2^k}\right)\geq 0.288$$

We can always repeat this process to increase the probability of success.

3.2 Complexity

The bound is not dependent on n, therefore only a constant number of trials is needed to increase the probability of success above any threshold, therefore the algorithm is in `BQP`.

It will with high probability perform $n-1$ subroutines each requiring $2n$ Hadamard operations and a call to function f, thus the complexity is $O(n^2\text{poly(n)})$.

Simon's algorithm was the first algorithm to separate `BPP` from `BQP`, it showed an exponential advantage as the best known classical algorithm is $O(2^{n/2}\text{poly(n)})$. The algorithm uses an oracle, therefore it does not have a practical application, however, it inspired others to seek more complex algorithms that can solve practical problems such as Shor's algorithm for prime factors.

4

Shor's Algorithm

Shor's Algorithm allows us to solve integer factorization problem in polynomial time using quantum computer. That is, given integer N the algorithm finds the set of all prime factors of N - prime number that divide N. Many attempts have been made to find a classical, polynomial algorithm with no apparent success, the best known classical solution is roughly $O(2^{(\log N)^{1/3}})$ [?], as a result Shor's algorithm is considered the strongest premise that BQP is not equal to BPP.

4.1 Reduction to order-finding

To achieve such an astounding reduction in complexity we have to reduce our problem of finding prime factors to a simpler problem. We easily see that to find all prime factors it is sufficient to be able to find a single non trivial prime factor, divide the original number by it and run the algorithm again. If we continue this procedure and store found factors we will solve the original problem as every number has a unique prime factorization.

We now want to reduce this problem further to the problem of order finding. For a number N we take a random value a from the set of $\{2, \ldots, N-1\}$. If we find the smallest number s such that

$$a^s = 1 \mod N$$

that is, find the period of a and assume that s is odd than we can rewrite it as

$$\begin{aligned} a^s - 1 &= 0 \mod N \\ (a^{s/2}+1)(a^{s/2}-1) &= 0 \mod N \\ (a^{s/2}+1)(a^{s/2}-1) &= kN \end{aligned}$$

now, if $a^{s/2} < N$ then we now that

$$N | (a^{s/2}-1)(a^{s/2}+1)$$

Note that from the condition $a^{s/2} < N$ one could assume that $a^{s/2}+1$ can be N, however this would imply that $a^{s/2} = 1 \mod N$, which contradicts the fact that s is the smallest number for which $a^s = 1 \mod N$. Therefore both $\gcd(a^{s/2} \pm 1, N) < N$.

From this we see that $\gcd(a^{s/2} \pm 1, N)$ are a factors of N, which we can compute classically, in polynomial time using Euclid's algorithm.

During the process we made two assumptions, s is even and $a^{s/2} \neq \pm 1 \mod N$. Both of those conditions are easily checked classically, we can also bound the probability of it not happening (s being valid) to be at least 1/4 [?]. If it happens, we simply choose different a and repeat the process.

We see that we reduced our factoring problem to the order finding problem - finding the order of a number in $\mathbb{Z}_N$. This problem itself is also NP-hard using classical approach, however we can exploit quantum Fourier transform to prove that it belongs to BQP.

4.2 Quantum Fourier Transform

To lower down the complexity of order-finding we will utilize a variant of Fourier Transform over the group $\mathbb{Z}_N$, where $N = 0^n$, which is a unitary operation over $\mathbb{C}^{2^n}$. We present a way to implement it using $O(n^2)$ quantum gates which is exponentially faster than a classical approach, using FFT would take $O(n2^n)$ [**?**]. However classical FFT outputs a vector from which we can read the result in its entirety, whereas QFT outputs an entangled quantum register that represents the same result, but upon measurement some information is lost. We cannot retrieve the full result, which means that QFT is only viable for certain applications. Luckily, one of them is order-finding.

The Quantum Fourier Transform is formally defined by the operation acting on a qubit belonging to the quantum register of length n, as

$$QFT(|x\rangle) = \frac{1}{\sqrt{N}} \sum_{y=1}^{N} e^{2\pi i x y/N} |y\rangle$$

We will compute it sequentially for each qubit, for i-th qubit we first apply a Hadamard gate on it to allow quantum interference, next we apply $n - i$ controlled phase gates to qubit i taking qubits $\{i + 1, n\}$ as control. Controlled phase gates take two qubits, target and control and apply a phase shift $e^{\pi/2^k}$ to the target qubit if the control qubit is in state $|1\rangle$, where k denotes the distance between qubits, we attach a visualization of the quantum circuit for 3 qubit system 2.

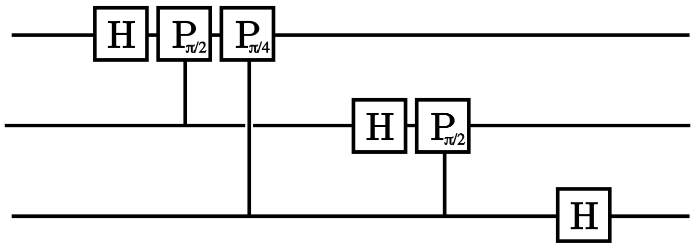

Figure 2: QFT diagram for an example, 3 qubit system

4.3 Order finding

we present a quantum algorithm that, given a and n such that $a < n, \gcd(a, n) = 1$ on input, finds r such that $a^r = 1 \mod n$. In other words, it finds the order of a in group F_n^*.

in the algorithm we use two quantum registers. first will hold the value $a^x \mod n$ and the second will store the exponent - x, initially both registers are set to ground state. As we need to utilize QFT, the registers' lengths need to be powers of 2 - $m = \lceil \log n \rceil$.

first we initialize the first register with m Hadamard gates, so it is in superposition of all possible x resulting in a state

$$\frac{1}{\sqrt{m}} \sum_{x=0}^{m-1} |x\rangle |1\rangle$$

then we compute the function $f(x) = a^x \bmod n$, and put the result in the second register resulting in

$$\frac{1}{\sqrt{m}} \sum_{x=0}^{m-1} |x\rangle \, |a^x \bmod n\rangle$$

then we measure the second register to get value y_0. as we know the value in the second register, value in the first register is restricted so that $a^x \mod n = y_0$. it can be thus written as $x_0 + lr$, where x_0 is the smallest x sufficing the condition. the current state is then

$$\frac{1}{\sqrt{\lceil m/r \rceil}} \sum_{l=0}^{\lceil m/r \rceil - 1} |x_0 + lr\rangle \, |y_0\rangle$$

then we apply QFT to the first register, to obtain the number x.

$$\frac{1}{\sqrt{m}\sqrt{\lceil m/r \rceil}} \sum_{x=0}^{m} \sum_{l=0}^{\lceil m/r \rceil - 1} \left[e^{2\pi i (x_0 + lr)x/m} \, |x\rangle \right] |y_0\rangle$$

the probability of obtaining certain x upon measurement is clearly proportional to $e^{2\pi(x_0+lr)x/m}$. from this we derive that probabilities will peak for values of x close to $\frac{m}{r}$,

we thus measure the value on the first register, resulting in x, in order to find r we need to classically find the best rational approximation for $\frac{x}{m}$, numbers a and b such that $\frac{x}{m} = \frac{a}{b}$. This can be done via the continued fractions algorithm [?]. the probability of x being close to $\frac{m}{r}$ is high, but not certain, we must check whether $a^b = a \mod m$, if so we output b, otherwise we can rerun the algorithm.

4.4 Bringing it together

In the algorithm we classically check if algorithm is prime, which can be done with AKS algorithm in deterministic, polynomial time [?]. We also compute greatest common divisor which can be solved using Euclid in algorithm $O(\log{(m+n)})$ [?].

We use the following subroutine recursively until N is prime. In each subsequent call we

- Check if N is not prime, if it is return it.
- Check if N is even, if it is return 2.
- Draw number a uniformly from $\{2, \ldots, N-1\}$.
- Compute the order s of a.
- Check if s is even and $y^{r/2} \neq \pm 1 \mod N$, if not go back to step 3.
- Compute $\gcd(a^{s/2} \pm 1, N)$ and return non-trivial factors.

We use the check $y^{r/2} \neq \pm 1 \mod N$ to ensure that we cannot find two non-trivial factors.

5

BQP and beyond

In analogue to BPP (Bounded probabilistic polynomial), BQP is defined as containing all problems solvable by a quantum computer with error probability of at most $c < 1/2$ for all problem instances.

5.1 BQP relations with classical complexity classes

We know that

$$\mathtt{BPP} \subseteq \mathtt{BQP} \subseteq \mathtt{PP} \subseteq \mathtt{PSPACE}$$

BPP

It is trivial to show that BQP contains BPP as a quantum computer can do all operations, as we can simply simulate a classical computer on a quantum computer. Shor's algorithm exponential reduction is strong evidence that in fact, BQP might be bigger than BPP, as it solves factoring, a well-studied problem efficiently. Unlike other problems that were thought to be not in BPP such as linear programming, we don't even have any heuristical solution that can produce correct output given some constraints.

NP

We don't know the relation between BQP and NP. Using Grover's search on quantum computers, we can get quadratic speed up of NP complete problems; however, this approach is not in BPP. It is widely believed that $\mathtt{NP} \nsubseteq \mathtt{BQP}$; however, we don't have any formal way to prove it.

PSPACE

We can show that using polynomial space and exponential time, we can simulate quantum computation, which implies that in fact $\mathtt{BQP} \subset \mathtt{PSPACE}$.

In order to simulate a T step quantum computation running on m-bit register, we need procedure $\mathtt{coeff}(x, i)$, which computes the coefficients of the quantum state at each step i given input x. Each quantum gate operation modifies at most 3 bits (or can be written as a combination of such), therefore to compute coeff on inputs x, i we need at most 8 recursive calls to $\mathtt{coeff}(x', i-1)$ as x and x' differ at most in 3 places. We can reuse the space used by each recursive call, therefore space used to calculate $\mathtt{coeff}(x, i)$ is at most space required to calculate $\mathtt{coeff}((x, i-1)$ plus space required to store the coefficient itself. If we denote space required to calculate $\mathtt{coeff}(x, i)$ as $S(i)$ and the number of bits in the coefficient as b, it can be written as

$$S(i) \leq S(i-1) + O(b)$$

Therefore $S(T)$ is polynomial.

To calculate the probability of getting a certain property, we just sum up the coefficients in $\mathtt{coeff}(x, T)$ for all x satisfying that property. Thus we can simulate quantum computation classically using polynomial space.

PP

PP is a complexity defined as containing all problems solvable in polynomial time, with error probability of less than 1/2 for all instances. It can be seen as similar to BPP with $c = 1/2$ (note that for BPP $c > 1/2$), meaning that problems in BPP are a subset of PP for which efficient algorithms exist.

This restriction on error probability makes it significantly larger than BPP, it contains both NP and coNP, moreover, it currently serves as the best upper bound for BQP[?]

5.2 Class QMA

QMA (Quantum Merlin Arthur) is the quantum analogue to NP. It contains all problems for which we can check the answer in quantum polynomial time. It trivially contains BQP, however, similarly to pair P, NP their exact relation is still unknown.

Quantum k-SAT

An example of a QMA-complete problem is a quantum k-SAT, $k > 2$. It is defined similarly to classical k-SAT.
As input it gets m hermitian operators $Q_1, \dots, Q_m$, each acting on k qubits out of total n qubits, the goal is to determine whether there exists an n-qubit quantum state $|\psi\rangle$ such that

$$\forall_i : Q_i |\psi\rangle = 0$$

Quantum 3-SAT has been proven to be QMA complete in 2013 by D. Gosset and DD. Nagaj [?]. From this result, it follows that k-SAT, $k > 2$ is QMA complete. It is worth noting that quantum 2-SAT can be solved in just polynomial time as shown by S. Bravyi in 2006 [?].

6

Discussion

While the core of this thesis is theoretical, it is worth discussing how quantum computing can be potentially utilized; the state of current advances in physical implementations and their limitations. As of 2025, relatively big, 100 state qubit machines are widely accessible for research centres and quantum-oriented companies worldwide. In theory, those machines should allow for the implementation of discussed algorithms effectively for small input; however, those machines are incredibly noisy, and any prolonged computation is close to impossible as the amount of noise generated renders the final output indistinguishable from noise.

Various implementations of quantum computers have emerged, each with their own benefits and drawbacks; this includes, among many: trapped ions, superconducting qubits, and photonic qubits, all of which can't overcome the toughest burden - quantum states are incredibly fragile. During computation, qubits are in a superposition of many states, and any disturbance, magnetic field, thermal oscillations, and even cosmic radiation can cause them to lose their quantum state; this problem is known as decoherence. Coherence time, which measures how long a qubit stays in its state before it gets disrupted, on current hardware is on the order of milliseconds. Additionally, quantum gates themselves are not error-free. Many attempts have been made for quantum error correction, none of which were successful enough to allow complex computation.

Despite those limitations, small instances of discussed algorithms were successfully implemented. Grover's search has been implemented for small 3-qubit instances [?]. Furthermore, Simon's algorithm, which demonstrates separation between classical and quantum complexity classes, has been demonstrated for 2-qubit instances [?]. Even factoring was successfully solved; nonetheless, the largest number factored using Shor's algorithm was only 21, back in 2012 [?]. Although there have been reports of factoring much larger numbers using different quantum approaches, all those attempts relied heavily on classical factoring, reducing the number of qubits and operations needed drastically; thus, they are widely recognized as just pretending to properly utilize quantum advantage [?]. Those results - although seemingly small in size, and not impressive - demonstrate feasibility, and the prospect that if hardware limitations are overcome, the now rather theoretical algorithms may prove useful.

In the past, most cryptographic algorithms, such as RSA and Diffie Hellman, based their security on factoring hardness. In theory then, Shor's algorithm can solve those problems effectively, which greatly contributed to the staggering amount of attention quantum computing has received in recent years. However, typical key sizes used with these algorithms are a few thousand bits long; therefore, with our current advances in factoring, we are likely decades away from any attempts to solve real-world ciphers using a quantum computer. Besides, as a result of this potential problem, cryptographers started to transition to algorithms that don't rely on integer factoring, so-called post-quantum algorithms.