Quantum Computation

Antoine Groudiev

December 21, 2023

One of the most widely used system for secure data transmission is the RSA scheme; its security relies on the assumption that the integer factorization problem is hard. To this day, there is indeed no known efficient algorithm to factor an integer *on a classical computer*.

Nevertheless, this problem can be solved in polynomial time on a quantum computer, using Shor's algorithm. A large enough quantum computer would therefore be able to break the RSA encryption scheme.

We will define the behavior of a quantum computer using two quantum computational models, and delve into quantum complexity theory, the study of complexity classes of problems solved using these quantum models. We will end by describing the Deutsch-Jozsa algorithm, to prove that a quantum computer can be exponentially faster than a deterministic classical computer.

Contents

1 Introduction to Quantum Computers **1**
- 1.1 Qubit representation . . . 2
  - 1.1.1 Dirac notation . . . 2
  - 1.1.2 Vector representation . . . 2
- 1.2 The Bloch sphere . . . 3

2 Quantum Computational Models **3**
- 2.1 Quantum Logic Gates . . . 3
- 2.2 Quantum Turing Machine . . . 3

3 Quantum Complexity Theory **3**
- 3.1 Quantum complexity classes . . . 3
  - 3.1.1 The BQP class . . . 3
  - 3.1.2 A BQP-complete problem . . . 3
  - 3.1.3 Relationship to classical complexity classes . . . 4
- 3.2 Relationship with the Church-Turing Thesis . . . 4

4 The Deutsch-Jozsa Algorithm **4**
- 4.1 Problem description . . . 4
- 4.2 Classical solution . . . 4
- 4.3 Quantum algorithm . . . 4

Conclusion **4**

1 Introduction to Quantum Computers

A classical computer uses the *bit* as a basic unit, a system which can be in exactly one of the two states `0` and `1`. Similarly, a quantum computer uses a basic unit called *qubit*, which can be

in a *superposition* of two states. A qubit can be described as a linear combinaison of both states, each coefficient being related to the probability that the qubit is in the given state. This way, a qubit can store more information than a simple bit, by storing information for both states at a time.

1.1 Qubit representation

While representing the state of a classical bit is quite simple, manipulating qubits requires more advanced notation.

1.1.1 Dirac notation

The state of a qubit is often represented using the Dirac notation, also known as the Bra-ket notation.

A qubit of state Ψ is noted $|\Psi\rangle$, and is in a superposition of the two state $|0\rangle$ and $|1\rangle$. This superposition can be written as:

$$|\Psi\rangle = \alpha|\Psi\rangle + \beta|\Psi\rangle$$

where $\alpha, \beta \in \mathbb{C}$ are called the states amplitudes.

Example (Schrödinger's cat)**.**

$$|\text{Schrödinger's cat}\rangle = \alpha|\text{dead}\rangle + \beta|\text{alive}\rangle$$

The probability to measure the qubit in the state $|0\rangle$ (and respectively $|1\rangle$) is $|\alpha|^2$ (respectively $|\beta|^2$). This interpretation gives us a normalization condition:

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

since the qubit must be mesured in one on the two states.

Example (LFCC exam)**.** *Assume that you have a probability $p \in [0, 1]$ of passing the next LFCC exam. Your result at the exam can be represented using one qubit in the following state:*

$$|\text{exam}\rangle = \sqrt{p}\,|\text{pass}\rangle + \sqrt{1-p}\,|\text{fail}\rangle$$

We can verify the normalization condition:

$$(\sqrt{p})^2 + (\sqrt{1-p})^2 = 1$$

1.1.2 Vector representation

Writing qubits is nice, but we would like to be able to change their state too. The Dirac notation is not the most practical one when the possible states ($|0\rangle$ and $|1\rangle$) are explicit. Instead, we can use a vector representation:

$$|\Psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$$

This notation allows us to use linear algebra tools to express physical conditions and changes. The normalization condition simply becomes:

$$\||\Psi\rangle\|_2 = 1$$

Furthermore, we can model an action that changes the qubit's state simply by a matrix U. If a qubit of initial state $|\Psi\rangle$ goes through some quantum gate (more on this later) represented by U, the final state $|\Psi'\rangle$ would be:

$$|\Psi'\rangle = U|\Psi\rangle$$

Obviously, we want $|\Psi'\rangle$ to respect the normalization condition for every qubit Ψ. This requires U to be a unitary matrix, that is $UU^\dagger = I_2$, where $U^\dagger$ is the conjugate transpose of U[1].

1.2 The Bloch sphere

2 Quantum Computational Models

To study the complexity of different problems with a quantum point of view, we need a model for a quantum computer. Much like classical computers, two approaches exists: the representation of a quantum computer as a circuit, using quantum logic gates, and an abstract model called quantum Turing machine, by analogy with the classical Turing machine.

2.1 Quantum Logic Gates

A quantum gate takes into input n qubits, and intuitively modifies the probability of each qubit to be in each state. More precisely, it changes the position of the qubit on the surface of the Bloch sphere. Therefore, quantum gates are often represented as unitary matrices of size $2^n \times 2^n$, i.e. elements of $U(2^n, \mathbb{C})$. We will present a few basic quantum gates.

2.2 Quantum Turing Machine

Another approach to model a quantum computer is to describe it as a generalized Turing machine. The set of states is replaced by a Hilbert space, and the transition function is replaced by a set of unitary matrices, similarly to the logic gates representation.

3 Quantum Complexity Theory

Similarly to the study of classical algorithms, we can regroup quantum problems in classes depending on their solvability by a quantum computational model under time and space constraints, such as time or space.

3.1 Quantum complexity classes

3.1.1 The BQP class

An important class of problem is the BQP class; it is the equivalent of the BPP class (Bounded-error, Probabilistic Polynomial time) for classical probabilistic computers.

Definition 1 (BQP)**.** *The BQP class is the set of problems which can be solved in polynomial time by a quantum Turing machine, with an error of at most* $\frac{1}{3}$*.*

The formal definition often involves a maximum error of $\frac{1}{3}$ in one run, but any bound strictly lower than $\frac{1}{2}$ is sufficient, since multiple runs of the algorithm can be chained to reduce the error.

3.1.2 A BQP-complete problem

Definition 2 (APPROX-QCIRCUIT-PROB problem)**.**

[1]The conjugate transpose is usually noted $U^\star$, but I guess that quantum physicists prefer the medieval vibe of †.

3.1.3 Relationship to classical complexity classes

3.2 Relationship with the Church-Turing Thesis

Definition 3 (Church-Turing Thesis)**.** *A function on the natural numbers can be calculated by an effective method if and only if it is computable by a Turing machine.*

Any quantum algorithm can be simulated either by hand, or on a classical computer, even if the resulting complexity of the computation is changed. Formally, it can be shown that a quantum Turing machine can be simulated on a Universal Turing Machine. Therefore, the existence of quantum computers do not disprove the Church-Turing thesis.

Nevertheless, the Church-Turing thesis do not state anything about the time complexity of the simulation. This lead to the following extension of the Church-Turing Thesis:

Definition 4 (Extended Church-Turing Thesis)**.** *A probabilistic Turing machine can efficiently simulate any realistic model of computation.*

Showing that $BPP \subsetneq BQP$ would imply that quantum computers are in certain cases more efficient than classical computers. This would therefore invalidate the extended Church–Turing thesis. A quantum-extended version of the thesis has therefore been proposed.

4 The Deutsch-Jozsa Algorithm

The Deutsch-Jozsa is an important quantum algorithm, not because of any practical use, but because it shows the interest of a quantum computer over a classical computer. The existence of the problem and algorithm proves that there is an *oracle separation* between the EQP and P class: this is a black box problem that can be solved in polynomial time using a quantum computer, but that cannot be solved by a classical computer in polynomial time.

4.1 Problem description

Let's consider a function f operating on n bits or qubits. That is:

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

This function is assumed to have a specific property: it is either *constant* or *balanced*:

$$|f^{-1}(\{0\})| = n \vee |f^{-1}(\{1\})| = n \vee \left(|f^{-1}(\{0\})| = |f^{-1}(\{1\})| = \frac{n}{2}\right)$$

This means that either the function produces only `0`, or it produces only `1`, or it produces exactly half of `0` and half of `1`.

The f function behaves like a black box, or an *oracle*: it is able to output the outcome of $f(x)$ is an single operation for any x. The computer is asked to determine whether the function is constant or balanced.

4.2 Classical solution

A deterministic computer needs in the worst case to mesure more than half the 2^n possible outcomes for f; the best time complexity is therefore exponential. Nevertheless, this problem can be solved efficiently on a probabilistic computer with a high probability.

4.3 Quantum algorithm

Conclusion