Quantum Computing

David Parker
CS 5720
University of Colorado Colorado Springs
Colorado Springs, CO USA
dparker@uccs.edu

***Abstract*—Quantum Computers have recently started to enter commercial production. This paper will review some of the basic concepts that make up a quantum computer and how it relates to computational complexity.**

***Keywords*—quantum computing, qubit, quantum gates, probability, np-complete**

I. Introduction

Modern computers use discrete values of 0 or 1 to represent information and logic gates built from transistors to process the information. In a quantum computer, the information is represented by physical states that they obey the laws of quantum mechanics. A quantum computer is a device able to manipulate and measure quantum states. The properties of quantum mechanics that make a quantum computer different then a classical computer are supposition and entanglement. Quantum computers use qubits to represent information and quantum gates to manipulate it.

II. Building Blocks of A Quantum Computer

A. Bits and Qubits

A bit in classical computers is represented by the states 0 or 1. It can manifest itself in many different forms including a hole in a punch card, the direction of a magnetic field, the reflectance of a surface or a voltage across a circuit. A qubit is two-state quantum version of a bit where the states can be 0, 1 or a combination of 0 and 1 at the same time. A qubit can also be physically manifested in many forms including the polarization of a photon, the atomic spin, and electron spin. The notation for representing a qubit in superposition is:

$c_0|0> + c_1|1>$

|0> and |1> are the basis states of the qubit. c_0 and c_1 are the probabilities that the system when measured will be 0 or 1 respectively where $|c_0|^2 + |c_1|^2 = 1$. c_0 and c_1 can be positive, negative, or even complex. When a qubit is measured it loses the probabilities and collapses into a state of 0 or 1 changing the state of the system.

A Qubit can be visualized using a Bloch Sphere. The Bloch Sphere is a sphere with a radius of one and a point on its surface represents the state of a qubit. The Bloch sphere can use angles to describe the state of a qubit. This representation allows any qubit state, including those with complex coefficients, to be represented as a point on the surface of the Bloch sphere.

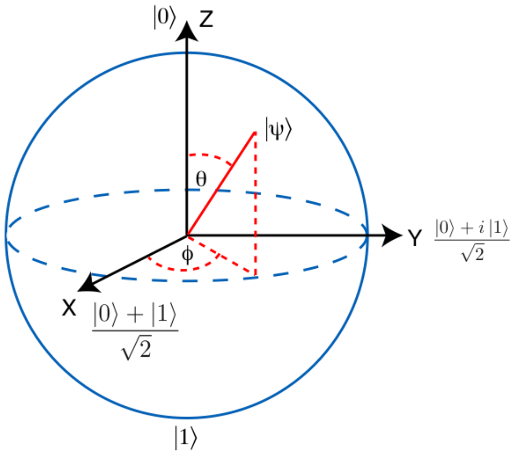
$|0\rangle$
Z
$|\psi\rangle$
θ
Y
$\frac{|0\rangle + i\,|1\rangle}{\sqrt{2}}$
ϕ
X
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
$|1\rangle$

Figure 1 Bloch Sphere

Another form of represent a qubit is with coordinates this can be useful when working with gate operation. In this form the basis states are $|0> = (1, 0)^T$ and $|1> = (0, 1)^T$.

Qubits may be combined to form a system of multiple quantum bits. For example a two qubit system is represented by the basis |00>, |01>, |10> and |11>. The state of the 2 qubit system is a unit-length vector: $c_0|00> + c_1|01> + c_2|10> + c_3|11>$ where $|c_0|^2 + |c_1|^2 + |c_2|^2 + |c_3|^2 = 1$.

B. Entanglement

Entanglement is a property of most quantum superpositions and does not occur in classical superpositions. In an entangled state, the whole system is in a definite state, even though the parts are not. Observing one of two entangled particles makes it behave randomly, but tells the observer exactly how the other particle would act if a similar observation were made on it. Because entanglement involves a correlation between individually random behaviors of the two particles, it cannot be used to send a message [1]. In an entangled state, (|00⟩+|11⟩)/2– √ , in which neither qubit has a definite state, even though the pair together does.

C. Gates

Gates perform the basic operations that a computer can perform on bits and qubits. Classical logic gates used in modern computing consist of the following operations on bits.

Table 1 Logic Gates

Gate	Input		Output	Gate	Input		Output
				NOT	0		1
					1		0
AND	0	0	0	NAND	0	0	1
	0	1	0		0	1	1
	1	0	0		1	0	1
	1	1	1		1	1	0
OR	0	0	1	NOR	0	0	1
	0	1	1		0	1	0
	1	0	1		1	0	0
	1	1	0		1	1	0
XOR	0	0	0	XNOR	0	0	1
	0	1	1		0	1	0
	1	0	1		1	0	0
	1	1	0		1	1	1

Note that NOR or NAND gates can reproduce the functionality of all the other gates.

Quantum gates are the basic building blocks of a quantum circuits. Quantum gates can leverage superposition and entanglement aspects of quantum mechanics that are not available to classical logic gates. One major difference between quantum gates and classical logic gates is the reversibility, that is you can always apply another gate to return the qubits to their original state. A unary quantum gate performs a unitary mapping on qubit $U: H_2 \to H_2$

Table 2 Unary Quantum Gates

Gate	Matrix	Description
Pauli X, NOT, bit flip	$\begin{matrix}0 & 1\\ 1 & 0\end{matrix}$	Transforms |0⟩ to |1⟩ and |1⟩ to |0⟩
Pauli Y	$\begin{matrix}0 & -i\\ i & 0\end{matrix}$	Transforms |0⟩ to i|1⟩ and |1⟩ to -i|0⟩
Pauli Z, Z, phase flip	$\begin{matrix}1 & 0\\ 0 & -1\end{matrix}$	No effect on |0⟩ but transforms |1⟩ to -|1⟩
Hadamard, H	$\frac{1}{\sqrt{2}}\begin{matrix}1 & 1\\ 1 & -1\end{matrix}$	Creates a superposition of the |0⟩ and |1⟩ states.
Phase Shift	$\begin{matrix}1 & 0\\ 0 & e^{i\emptyset}\end{matrix}$	Modifies the phase of the quantum state without changing the probability of measuring a |0> or |1>.
Phase, $\frac{\pi}{4}$, S	$\begin{matrix}1 & 0\\ 0 & i\end{matrix}$	
$\frac{\pi}{8}$	$\begin{matrix}1 & 0\\ 0 & e^{i\frac{\pi}{4}}\end{matrix}$	

Quantum gates can be defined to operate on multiple qubits. Using a Tensor product of two quantum gates generates a gate that is equal to the two gates in parallel.

$$H \otimes H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \otimes \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{bmatrix}$$

When a two-qubit parallel hadamard gate is applied to |00> it creates a quantum state that have equal probability of being observed in any of its four possible outcomes; 00, 01, 10 and 11.

$$\frac{1}{2}\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{bmatrix}\begin{bmatrix}1\\ 0\\ 0\\ 0\end{bmatrix} = \frac{1}{2}\begin{bmatrix}1\\ 1\\ 1\\ 1\end{bmatrix} = \frac{1}{2}|00> + \frac{1}{2}|01> + \frac{1}{2}|10> + \frac{1}{2}|11>$$

Table 3 Quantum Gates

Gate	Matrix	Description
SWAP	$\begin{matrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{matrix}$	Swaps the two input qubits around.
CNOT	$\begin{matrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{matrix}$	Performs the NOT operation on the second qubit only when the first qubit is |1⟩
CCNOT, Toffoli	$\begin{matrix}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\end{matrix}$	If the first two bits are in the state |1> , it applies a Pauli-X (or NOT) on the third bit, else it does nothing.
CSWAP, Fredkin	$\begin{matrix}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\end{matrix}$	Performs a controlled swap

III. Circuits

A quantum circuit on m qubits is a unitary mapping on H_{2m}, which can be represented as a combination of a finite set of quantum gates [2]. All quantum circuits and be constructed using either the controlled not gate and unary gates or the Toffoli gates and Hadamard-Walsh gates. The quantum circuit will perform operations on a set of qubits and modify the probabilities of the qubits being measured as |0> or |1>. The measurement of the typically on occurs after all the operations have been carried out because intermediate measurements could destroy the states of the qubits.

IV. Quantum Turing Machines

To provide a conceptual model of quantum computation a Non-deterministic Turing machine can be modified to add transition amplitudes to the transition functions. Quantum circuits are preferred over quantum turning machines for representing quantum algorithms.

V. Complexity

Two quantum complexity classes are BQP and QMA which are the quantum analogues of P and NP.

BQP (bounded-error quantum polynomial time) is the class of decision problems solvable by a quantum computer in polynomial time, with an error probability of at most 1/3 for all instances. A decision problem is a member of BQP if there exists an algorithm for a quantum computer (a quantum algorithm) that solves the decision problem with high probability and is guaranteed to run in polynomial time. A run of the algorithm will correctly solve the decision problem with a probability of at least 2/3.

QMA (Quantum Merlin Arthur) Is the set of decision problems for which there is a BQP verifier.

A. NP-Complete with Quantum Computers

While it is tempting to think that a quantum computer would be able to solve all NP-Complete problems in polynomial time, that is not necessarily the case. Suppose you're searching a space of 2^n possible solutions for a single valid one, and suppose that all you can do, given a candidate solution, is feed it to a 'black box' that tells you whether that solution is correct or not. Classically, it's clear that you need to query it 2^n times in the worst case. On the other hand, Grover gave a quantum search algorithm that queries the black box only $2^n/2$ times. In other words, any quantum algorithm to find a needle in a size-2^n haystack needs at least $2^n/2$ steps [2]. This means that for "generic" search problems, quantum computers can give a quadratic speedup.

There are some problems that can benefit from an exponential speedup such as Shor's factoring algorithm.

VI. References

[1] IBM Research and the IBM QX team, "IBM Q Experience Users Guide," IBM, 2017. [Online]. Available: https://quantumexperience.ng.bluemix.net/qx/tutorial. [Accessed 2018].

[2] M. Hirvensalo, "Quantum Computing," in *Quantum Computing 2nd*, Springer Publishing Company, Incorporated, 2010.

[3] S. Aaronson, "PHYS771 Lecture 10: Quantum Computing," 2006. [Online]. Available: https://www.scottaaronson.com/democritus/lec10.html.