CS333 - Problem Set 8

1. In this problem, we will investigate a many-qubit version of Deutsch's Algorithm called the Deutsch-Josza Algorithm. This problem involves the unitary operation $H^{\otimes n}$, so we will first investigate its properties. $H^{\otimes n}$ is n copies of H acting simultaneously on n qubits, and can be described using a circuit diagram as:

$$\begin{array}{c} -\boxed{H}- \\ -\boxed{H}- \\ -\boxed{H}- \end{array} \tag{1}$$

(The above circuit in particular describes $H^{\otimes 3}$) . There is a concise way to describe how $H^{\otimes n}$ acts on standard basis states of n qubits. Let $|x\rangle$ be a standard basis state on n qubits. This means $|x\rangle$ can be written as a tensor product of n standard single qubit basis states, i.e. $|x\rangle = |x_1 x_2 x_3 \dots x_n\rangle$ where each x_i is either 0 or 1. Then

$$H^{\otimes n}|x\rangle = \frac{1}{2^{n/2}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle. \tag{2}$$

where the summation is over all 2^n standard basis states y, and

$$x \cdot y = \sum_{i=1}^{n} x_i y_i \tag{3}$$

where x_i is the ith bit of x and y_i is the ith bit of y.

So for example:

$$\begin{aligned} H^{\otimes 2}|01\rangle &= \frac{1}{2} \sum_{y \in \{0,1\}^n} (-1)^{01 \cdot y} |y\rangle \\ &= \frac{1}{2} \left((-1)^{01 \cdot 00}|00\rangle + (-1)^{01 \cdot 01}|01\rangle + (-1)^{01 \cdot 10}|10\rangle + (-1)^{01 \cdot 11}|11\rangle \right) \\ &= \frac{1}{2} \left((-1)^{0 \times 0 + 1 \times 0}|00\rangle + (-1)^{0 \times 0 + 1 \times 1}|01\rangle + (-1)^{0 \times 1 + 1 \times 0}|10\rangle + (-1)^{0 \times 1 + 1 \times 1}|11\rangle \right) \\ &= \frac{1}{2} \left(|00\rangle - |01\rangle + |10\rangle - |11\rangle \right). \end{aligned} \tag{4}$$

(a) Use the formula in Eq. (2) to determine the resulting state of

$$H^{\otimes n}|0\rangle^{\otimes n} \tag{5}$$

and

$$H^{\otimes n}|1\rangle^{\otimes n}. \tag{6}$$

($|0\rangle^{\otimes n}$ is the n-qubit standard basis state where $x_1 = x_2 = \cdots = x_n = 0$ and $|1\rangle^{\otimes n}$ is the n-qubit standard basis state where $x_1 = x_2 = \cdots = x_n = 1$.)

(b) (Challenge question) Let $|x\rangle$ be a standard basis state that is not $|0\rangle^{\otimes n}$. Explain why

$$H^{\otimes n}|x\rangle \tag{7}$$

produces a superposition of all standard basis states with exactly half of the amplitudes positive and half of the amplitudes negative.

(c) Consider a function $f : \{0,1\}^n \to \{0,1\}$ (f takes as input an n-bit string and outputs 1 bit) that has the promise that it is either even or balanced. If it is even, $f(x) = 0$ for all inputs, or $f(x) = 1$ for all inputs. If it is balanced, then for exactly half of the inputs, $f(x) = 0$ and for half of the inputs $f(x) = 1$. We would like to determine which case we are in.

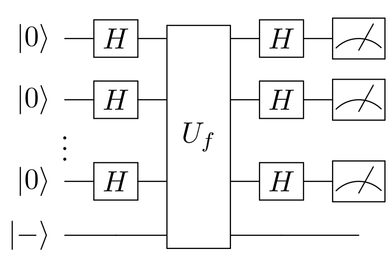
i. What is the worst case deterministic classical query complexity of deciding with certainty if the function is even or balanced? You should assume that you have to make public your algorithm ahead of time, before getting access to f. Then an adversary gets to design f in such a way to try to make your algorithm make the maximum number of queries possible. The query complexity is the maximum number of queries in this setting.

ii. What is the worst case probabilistic classical query complexity of determining if the function is even or balanced? This means you have a probabilistic algorithm, and your answer should be correct with probability at least 2/3. You don't have to prove your algorithm's query complexity (although you can), just sketch the idea. You should assume that you have to make public your algorithm ahead of time, before getting access to f. Then an adversary gets to design f in such a way to try to make your algorithm make the maximum number of queries possible. The query complexity is the maximum number of queries in this setting. Why is this case different from the deterministic setting?

(d) If $|x\rangle$ is a standard basis state on n qubits and $|y\rangle$ is a standard basis state on 1 qubit, then U_f acts as follows:

$$U_f|x\rangle_A|y\rangle_B = |x\rangle_A|y \oplus f(x)\rangle_B, \tag{8}$$

where A is an n qubit system and B is a 1 qubit system. Consider the following circuit



(9)

i. What is the state of the system after U_f acts?

ii. What is the state of the system after the second $H^{\otimes n}$ acts?

iii. If the function is even, what is the probability of getting outcome $|0\rangle^{\otimes n}$ (i.e. $|0\rangle$ on every qubit) when the measurement is made? What if the function is balanced?

iv. What is the quantum query complexity of determining if the function is even or balanced?

(e) Please comment on the difference/similarity between deterministic, probabilistic, and quantum query complexity and why these differences/similarities occur. Use the word "interference" in your answer.

2. (a) Suppose you know that over the course of a quantum algorithm on n qubits, the quantum system is never in more than a superposition of T standard basis states. Suppose that the computation involves M single and two qubit unitaries. You may assume that at each time step, only one unitary acts at a time (so there is no parallel computation). Explain how you can simulate this computation using a classical computer with $O(Tn)$ bits in $O(MT^2n)$ time. (You can probably do $O(MTn)$ time with some extra cleverness...bonus points if you do!)

(b) (Big picture) Based on the previous part, what property of a quantum algorithm is necessary (although not sufficient) in order to have an exponential speed-up versus any classical algorithm?

3. Quantum Hype Chapter I: Superposition

(a) 3-SAT is the problem of determining whether, given a Boolean formula of the form

$$f(x_1, x_2, \dots, x_n) = (x_1 \vee \neg x_2 \vee x_3) \wedge (x_2 \vee x_4 \vee \neg x_5) \wedge \cdots \wedge (\neg x_3 \vee \neg x_4 \vee x_n), \quad (10)$$

there is an assignment of each variable x_i to either 1 (TRUE) or 0 (FALSE) so that the whole formula has value 1 (TRUE). We call such an assignment a satisfying assignment. This problem is NP-Complete, which means most people believe it takes exponential time (in n) to solve using a classical computer. (In other words, to solve you basically have to test all 2^n possible ways of assigning 0 and 1 to the variables.)

Many important real world problems (for example the traveling salesperson problem) are equivalent to 3-SAT, which means that if you can come up with an improved algorithm for 3-SAT, you can also solve a lot of other useful but hard problems in less than exponential time. Some people are hopeful that quantum computers can solve NP-Complete problems like 3-SAT. This hope stems from the fact that you can evaluate $f(x_1, x_2, \dots, x_n)$ on all 2^n possible assignments using only $n+1$ qubits and polynomial time.

To see this, let $U_f|x_1, x_2, \dots, x_n\rangle|y\rangle = |x_1, x_2, \dots, x_n\rangle|y \oplus f(x_1, x_2, \dots, x_n)\rangle$. U_f can be implemented in polynomial time given a description of f. Note that this is a different scenario than in Deutsch's algorithm, where we don't have a description of f, and instead only have access to U_f. In this case, we know all of the details of f, but it is still very hard to figure out whether a satisfying assignment exists.

i. Analyze the output state of the following circuit: $U_f(H^{\otimes n} \otimes I)|0\rangle^{\otimes n+1}$

ii. Explain how the output state contains information about whether there is a satisfying assignment.

iii. Despite the state containing the information needed to solve this very hard problem, explain why you can't actually access this information, and explain what happens when you try to measure and extract the information.

(b) Explain what is problematic about the following descriptions of quantum computing, which are typical examples of how superposition and quantum algorithms are described in various types of media:

- Read bullet 2 of "In Brief" in this Acccenture report(?) [someone who knows about business things please tell me the appropriate word to describe this type of write-up].
- Read the first three paragraphs of Quantum computers may be more of an imminent threat than AI by Vivek Wadhwa, originally published in the Washington Post in 2018.

(c) Explain this tweet.