Quantum Computing 2/17/2022

Lecture 8: Phase Kickback

Instructor: Dieter van Melkebeek

In today's lecture, we primarily focus on solving blackbox problems using quantum algorithms. We first define the quantum blackbox problems and our objective. We then introduce the "phase kickback" technique which is a foundation of the algorithms we discuss next: determining whether the blackbox function is constant or balanced, learning linear functions.

1 Blackbox algorithms

Classical blackbox problems In blackbox problems, we are given a blackbox function $f : \{0,1\}^n \to \{0,1\}^l$ and expected to determine some property of it. In doing so, we can make some queries to the blackbox. In a single query, we select any input in $\{0,1\}^n$ and receive the result of applying f to our input.

Quantum blackbox problems In the quantum version, we are instead given an access to $f : \{0,1\}^n \times \{0,1\}^l \to \{0,1\}^n \times \{0,1\}^l$ because a valid quantum computation must be a unitary transformation. We denote this transformation as $U_f : |x\rangle |y\rangle \mapsto |x\rangle |y \oplus f(x)\rangle$. Note that U_f is also its own inverse. Now, the input and output of the query is in the form of $\{0,1\}^n \times \{0,1\}^l$.

Query complexity The query complexity is the number of applications of blackbox needed in any algorithm for the problem. One of the main goals in developing blackbox algorithms is to achieve this minimum.

A few caveats – Query complexity does not take into account non-query operations. In other words, we don't consider how fast the algorithm is including all other operations besides the calls to the blackbox. We also ignore how to implement U_f which may not be trivial in some problems such as search problem. Finally, when comparing quantum to classical query complexity numerically we are implicitly ignoring the difference in query mechanism.

On the other hand, many tight bounds are known for both classical and quantum query complexity. Those allow us to demonstrate gaps between the power of deterministic, probabilistic, and quantum blackbox algorithms.

2 Phase kickback

Consider a single-output function $f : \{0,1\}^n \to \{0,1\}$, i.e., we set $l = 1$. We want to design a quantum circuit that realizes the unitary mapping $|x\rangle \mapsto (-1)^{f(x)} |x\rangle$ for every $x \in \{0,1\}^n$. We call this transformation that encodes $f(x)$ into the phase of $|x\rangle$ a "phase kickback". We present two algorithms that each use a single ancilla qubit whose state is unaffected. The state $|\phi\rangle$ of the ancilla differs between the two.

Two queries with $|\phi\rangle = |0\rangle$ The algorithm is built on the observation that $Z|0\rangle = |0\rangle$ and $Z|1\rangle = (-1)|1\rangle$ where Z is a Pauli-Z gate. Hence, applying Z to the ancilla qubit of the state $|x\rangle|f(x)\rangle$ resulted in $(-1)^{f(x)}|x\rangle|f(x)\rangle$.

The algorithm utilizes this observation as follow. Starting from a state $|x\rangle|0\rangle$, the state immediately after applying U_f is $|x\rangle|f(x)\rangle$. We then apply Z to the ancilla qubit to get $(-1)^{f(x)}|x\rangle|f(x)\rangle$. Finally, applying U_f again will revert the ancilla qubit making it be $(-1)^{f(x)}|x\rangle|0\rangle$ which is what we want our final state to be.

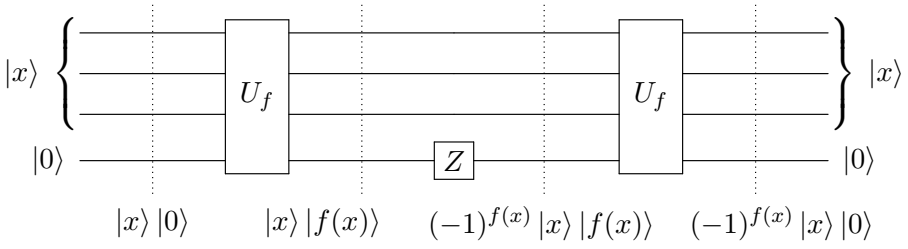

Figure 1: Two queries with $|\phi\rangle = |0\rangle$

Single query with $|\phi\rangle = |-\rangle$ Recall that $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, hence $|x\rangle|-\rangle = \frac{1}{\sqrt{2}}(|x\rangle|0\rangle - |x\rangle|1\rangle)$. Starting with this state, we apply U_f to it to get $\frac{1}{\sqrt{2}}(|x\rangle|f(x)\rangle - |x\rangle|\overline{f(x)}\rangle)$. The state reduces to $|x\rangle|-\rangle$ when $f(x) = 0$ and to $(-1)|x\rangle|-\rangle$ when $f(x) = 1$. Therefore, we can achieve the state $(-1)^{f(x)}|x\rangle|-\rangle$ using only one query to U_f.

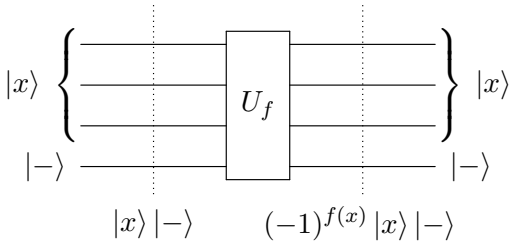

Figure 2: Single query with $|\phi\rangle = |-\rangle$

3 Distinguishing constant from balanced

Consider a function $f : \{0,1\}^n \to \{0,1\}$ where f can be either "constant" or "balanced". We call f constant if it is identically 0 or 1. We call f balanced if the number of 0's and 1's that f maps to is equal. We want to design a quantum algorithm to determine whether the blackbox function is constant or balanced.

Promise problems Note that the problem we are considering here is only partially specified. In particular, the output is not specified in cases where f is neither constant nor balanced. For $n = 1$

there are no such cases, but there are for $n > 1$. Such partially specified problems are often called "promise problems," where the term "promise" refers to the fact that the input to the problems is guaranteed to satisfy some property that not all possible inputs (necessarily) satisfy. In this case the promise is that "f is either constant or balanced". For $n = 1$ this includes all possible functions, but not for $n > 1$.

The difference between promise problems and fully specified problems is important in query complexity. For promise problems exponential gaps are known between quantum query complexity and classical query complexity. For fully specified problems only polynomial gaps are possible. In modern lingo, for promise problems quantum supremacy is possible in terms of query complexity, whereas for fully specified problems only quantum advantage can be achieved.

3.1 Deutsch algorithm: $n = 1$

Recall that, if we apply phase kickback to states $|0\rangle|-\rangle$ and $|1\rangle|-\rangle$, we get $(-1)^{f(0)}|0\rangle|-\rangle$ and $(-1)^{f(1)}|1\rangle|-\rangle$ respectively. Thus, if we apply this phase kickback technique to the superposition state $|+\rangle|-\rangle$, we have

$$U_f|+\rangle|-\rangle = \frac{1}{\sqrt{2}}(U_f|0\rangle|-\rangle + U_f|1\rangle|-\rangle) = \frac{1}{\sqrt{2}}((-1)^{f(0)}|0\rangle|-\rangle + (-1)^{f(1)}|1\rangle|-\rangle)$$

Notice that, if $f(0) = f(1)$, the state ends up being $(-1)^{f(0)}|+\rangle|-\rangle = \pm|+\rangle|-\rangle$. On the other hand, if $f(0) \neq f(1)$, the state reduces to $(-1)^{f(0)}|-\rangle|-\rangle = \pm|-\rangle|-\rangle$. Consider the two possible states of the non-ancilla qubit $|+\rangle$ and $|-\rangle$, these states are orthogonal. Hence, we can find a transformation that maps one of them to $|0\rangle$ and the other to $|1\rangle$. Such transformation is the Hadamard gate which maps $|+\rangle$ to $|0\rangle$ and $|-\rangle$ to $|1\rangle$. This mapping allows us to determine the property of f by measuring the non-ancilla qubit which can either be $|0\rangle$ or $|1\rangle$ with no error.

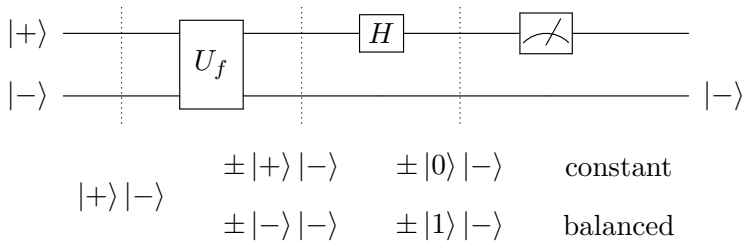

Figure 3: Deutsch algorithm

3.2 Deutsch-Jozsa algorithm: $n > 1$

This algorithm also utilizes the phase kickback technique like we did in 3.1. Now, we start with the state $|\psi\rangle = |+\rangle|+\rangle\ldots|+\rangle|-\rangle$. We also denote $|\psi\rangle$ as

$$|\psi\rangle = \left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right)\ldots\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right)|-\rangle = \frac{1}{\sqrt{2^n}}\left(\sum_{x\in\{0,1\}^n}|x\rangle\right)|-\rangle$$

Applying a phase kickback technique U_f to $|\psi\rangle$, we have

$$U_f |\psi\rangle = \frac{1}{\sqrt{2^n}} \left(\sum_{x \in \{0,1\}^n} (-1)^{f(x)} |x\rangle \right) |-\rangle$$

Notice that if f is constant, the state reduces to

$$U_f |\psi\rangle = \frac{1}{\sqrt{2^n}} \left(\sum_{x \in \{0,1\}^n} (-1)^{f(0)} |x\rangle \right) |-\rangle = \pm |+\rangle |+\rangle \dots |+\rangle |-\rangle$$

The state $U_f |\psi\rangle$ has a more complicated form when f is not constant. Still, we know that, if f_1 is any constant function and f_2 is any balanced function, the state of non-ancilla qubits of $U_{f_1} |\psi\rangle$ is orthogonal to that of $U_{f_2} |\psi\rangle$. This can be proven by taking the inner product. Since f_2 is balanced, $\sum_x (-1)^{f_2(x)} = 0$, we have the inner product of f_1 and f_2 $\sum_x (-1)^{f_1(0)} \cdot (-1)^{f_2(x)} = 0$ making them orthogonal. Hence, there exists a transformation that maps the only state when f is constant, $|+\rangle^{\otimes n}$, to the known basis $|0^n\rangle$ and the states when f is balanced to states with no component along $|0^n\rangle$.

Such transformation is, again, a Hadamard gate. Applying the gate to each of the non-ancilla qubits, we then observe $|0^n\rangle$ if and only if f is constant. ($H |+\rangle = |0\rangle$). If we observe anything else, then f must be balanced. This mapping allows us to determine the property of f with no error with only one query.

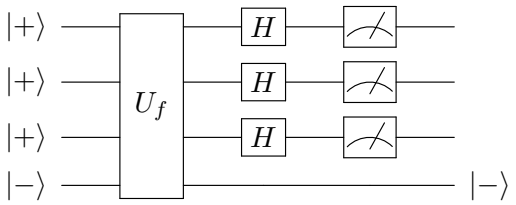

Figure 4: Deutsch-Jozsa algorithm

Note that we can achieve $|+\rangle$ from applying Hadamard gate to $|0\rangle$. Therefore, our circuit may look like the following Figure 5 where $H^{\otimes n}$ denotes a Hadamard tensor gate which is applying Hadamard gate to each qubit.

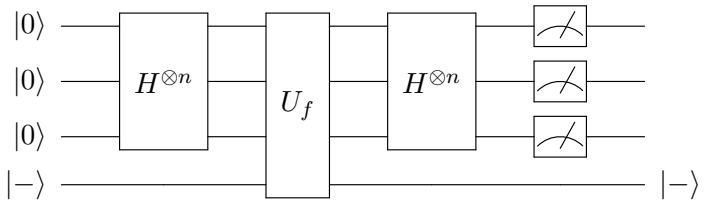

Figure 5: Deutsch-Jozsa algorithm

Query complexity In general, if we want to determine the property deterministically in classical computation, we need $N/2+1$ queries where $N = 2^n$. This is because if we make only $N/2$ queries and they are all 0, we don't know whether the underlying function f is identically zero or balanced; the next query will determine it for sure.

We can also consider a probabilistic algorithm that uses k queries to find the property with error $\leq 1/2^{k-1}$. Pick the queried uniformly at random. If the values are not the same, then the function is balanced. Otherwise, we can't conclude the property for sure. Still, we can argue that the function is balanced with probability $(1/2)^{k-1}$ since, if f is balanced, for each query after the first the probability that it returns the same value as the first is $1/2$, and these events are independent. This simple probabilistic algorithm is pretty much the best one can do in that setting, as the following exercise shows.

Exercise 1. *Show that every probabilistic algorithm with k queries has error $\Omega(1/2^k)$. Hint: use Yao's Principle.*

In contrast, our quantum algorithm solves the problem exactly with a single query.

4 Hadamard tensor

The Hadamard tensor $H^{\otimes n}$ is the operation of n Hadamard gates on n qubits state. We already saw that we can create a uniform superposition on n qubits by applying $H^{\otimes n}$ to $|0^n\rangle$.

$$H^{\otimes n} |0^n\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} |y\rangle$$

What if we apply $H^{\otimes n}$ to a generic basis state $|x\rangle$ for $x = x_1 x_2 \dots x_n \in \{0,1\}^n$?

$$\begin{aligned}
H^{\otimes n} |x\rangle &= H|x_1\rangle \, H|x_2\rangle \dots H|x_n\rangle \\
&= \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_1}|1\rangle) \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_2}|1\rangle) \dots \frac{1}{\sqrt{2}}(|0\rangle + (-1)^{x_n}|1\rangle) \\
&= \frac{1}{\sqrt{2}} \left(\sum_{y_1=0}^{1} (-1)^{x_1 y_1} |y_1\rangle \right) \frac{1}{\sqrt{2}} \left(\sum_{y_2=0}^{1} (-1)^{x_2 y_2} |y_2\rangle \right) \dots \frac{1}{\sqrt{2}} \left(\sum_{y_n=0}^{1} (-1)^{x_n y_n} |y_n\rangle \right) \\
&= \frac{1}{\sqrt{N}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle , \qquad (1)
\end{aligned}$$

where $x \cdot y \doteq \sum_{i=1}^{n} x_i y_i$.

5 Learning linear functions

Consider a promise function $f : \{0,1\}^n \to \{0,1\}$ where $f(x) = a \cdot x \bmod 2$ for some $a \in \{0,1\}^n$. Our goal is to find a using blackbox access to f.

The quantum circuit for this problem is exactly the same for the one for distinguishing constant from balanced, but the final observation is going to be our a. Since $f(x) = a \cdot x \bmod 2$, we can

rewrite $(-1)^{f(x)} = (-1)^{a\cdot x \bmod 2} = (-1)^{a\cdot x}$. Thus, the state after applying U_f is

$$\frac{1}{\sqrt{2^n}} \sum_x (-1)^{f(x)} |x\rangle |-\rangle = \frac{1}{\sqrt{2^n}} \sum_x (-1)^{a\cdot x} |x\rangle |-\rangle . \tag{2}$$

Comparing the right-hand side of (2) with (1), we can rewrite (2) as $H^{\otimes n} |a\rangle |-\rangle$. As the Hadamard gate is its own inverse, apply $H^{\otimes n}$ to the first n qubits yields the state $|a\rangle |-\rangle$. Measuring the first n qubits of this state yields a with certainty. Thus, we can find a exactly with a single quantum query.

The resulting quantum algorithm is commonly used as a benchmark for quantum computers. To do so, the blackbox needs to be realized, for a given a, in terms of elementary gates, as well as the other operations. The following exercise shows how it can be done efficiently.

Exercise 2. *Fix $a \in \{0,1\}^n$.*

(a) Implement U_f for $f(x) = a \cdot x \bmod 2$ using CNOTs only.

(b) Show that $H^{\otimes 2} \circ CNOT \circ H^{\otimes 2}$ is equivalent to a CNOT with the control swapped.

(c) Use (b) to reduce the number of elementary gates in the resulting quantum circuit.

Query complexity We can deterministically find a using n queries by finding $f(100\ldots0)$, $f(010\ldots0)$, ..., $f(0\ldots01)$ which will tell us a_1, a_2, ..., a_n. Note that n queries is optimal since we can retrieve at most one bit of information from each classical query, and there are 2^n possibilities for a. Also, every probabilistic algorithm with error less than $1/2$ needs to make at least n queries. We leave the proof as an exercise. Our quantum algorithm improves the query complexity to just one query with no error.

Exercise 3. *Show that every probabilistic algorithm with $n-1$ queries has error at least $1/2$. Hint: use Yao's Principle.*