Introduction to Quantum Computation
Summer 2020, Paderborn University

Sevag Gharibian

Lecture 7: The Deutsch-Josza and Berstein-Vazirani algorithms

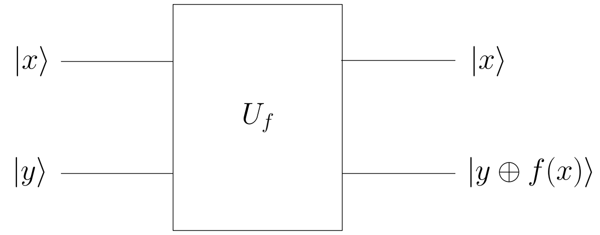
> *"Computers are physical objects, and computations are physical processes. What computers can or cannot compute is determined by the laws of physics alone. . ."*
> — David Deutsch

> *"Where there's smoke, there's fire."*
> — Latin proverb

In the lectures thus far, we've introduced the postulates of quantum mechanics, and studied them through the lens of quantum information theoretic concepts such as entanglement, non-local games, and entropy. We now switch to the theme of *quantum algorithms*, i.e. algorithms harnessing the four postulates of quantum mechanics. We begin with a simple quantum algorithm due to David Deutsch, which is by no means new (it was discovered in 1985), nor does it tackle a particularly important problem (in fact, the problem is quite artificial). Nevertheless, Deutsch's algorithm serves as an excellent proof of concept that, in certain settings, quantum computers are strictly more powerful than classical ones. Moreover, as shown by Berstein and Vazirani, the generalization of this algorithm (dubbed the Deutsch-Josza algorithm) can actually be seen as solving a more natural problem — given a black-box function $f : \{0,1\}^n \mapsto \{0,1\}$ which computes $f(x) = a \cdot x \mod 2$ for some unknown $a \in \{0,1\}^n$, what is a?

1 The setup: Functions as oracles

The problem which Deutsch's algorithm tackles is stated in terms of binary functions $f : \{0,1\} \mapsto \{0,1\}$. Thus, the first thing we'll need to do is understand how to *model* such functions in the quantum circuit model. What makes the task slightly non-trivial is that, recall by Postulate 2 of quantum mechanics, all quantum operations must be unitary and hence *reversible*. In general, however, given the output $f(x)$ of a function, it is not always possible to *invert* f to obtain the input x. In other words, we have to compute $f(x)$ in such a way as to guarantee that the computation can be undone. This is achieved via the following setup:



Here, $U_f \in \mathcal{U}((\mathbb{C}^2)^{\otimes 2})$ is a unitary operator mapping $|x\rangle|y\rangle \mapsto |x\rangle|x \oplus y\rangle$ for any $x, y \in \{0,1\}$ (i.e. $|x\rangle, |y\rangle$ denote standard basis states), and where $\oplus$ denotes XOR or addition modulo 2. Note that by linearity, once we define the action of U_f on standard basis states, we immediately know how it acts on *any* input state $|\psi\rangle \in (\mathbb{C}^2)^{\otimes 2}$.

Exercise. Let $|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$. What is the state $U_f|\psi\rangle$?

Observe now that U_f is reversible — this is because running U_f again on its output, $|x\rangle|y \oplus f(x)\rangle$, yields state $|x\rangle|y \oplus f(x) \oplus f(x)\rangle = |x\rangle|y\rangle$, since $f(x) \oplus f(x) = 0$ (adding the same bit twice and dividing by 2 leaves remainder zero). Second, note that we have not specified the inner workings of U_f (i.e. we have not given a circuit implementing the functionality stated above); in this course, we shall treat U_f as a "black box" or "oracle" which we presume we can run, but cannot "look inside" to see its implementation details.

2 The problem: Is f constant or balanced?

The problem Deutsch's algorithm tackles can now be stated as follows. Given a block box U_f implementing some unknown function $f : \{0,1\} \mapsto \{0,1\}$, determine whether f is "constant" or "balanced". Here, *constant* means f always outputs the same bit, i.e. $f(0) = f(1)$, and *balanced* means f outputs different bits on different inputs, i.e. $f(0) \neq f(1)$.

Exercise. Suppose $f(0) = 1$ and $f(1) = 0$. Is f constant or balanced? Given an example of a constant f.

Of course, there is an easy way to determine whether f is constant or balanced — simply evaluate f on inputs 0 and 1, i.e. compute $f(0)$ and $f(1)$, and then check if $f(0) = f(1)$. This naive (classical) solution, however, requires two *queries* or calls to U_f (i.e. one to compute $f(0)$ and one to compute $f(1)$). So certainly at most two queries to U_f suffice to solve this problem. Can we do it with just *one* query? Classically, the answer turns out to be no. But quantumly, Deutsch showed how to indeed achieve this with a single query.

Quantum query complexity. As you may have noticed above, the "cost function" we are interested in minimizing in solving Deutsch's problem is the number of quantum queries to U_f. This is an example of the model of *quantum query complexity*, in which many quantum algorithms have been developed. In the study of quantum query complexity, one is given a black box U_f implementing some function f, and asked what the minimum number of required queries to U_f is in order to determine some desired property of f. Note that the quantum algorithm computing this property can consist of (say) 999999999 quantum gates; if it contains only 2 queries to U_f, then we consider the cost of the algorithm as 2, i.e. all "non-query" operations are considered free.

3 The algorithm

3.1 A naive idea

Before demonstrating the algorithm itself, let us first attempt a simpler, more naive approach — since we are allowed to query U_f quantumly, what happens if we just query U_f in *superposition* in the input register? In other words, what happens if we run the circuit for U_f with input state $|x\rangle$ replaced with $\alpha|0\rangle + \beta|1\rangle$ and output state $|y\rangle$ with $|0\rangle$? Intuitively, here we have set the input register to *both* possible inputs 0 and 1, and so we expect U_f to return a superposition of both possible outputs, $f(0)$ and $f(1)$. Indeed, by linearity of U_f, the output of the circuit will be

$$|\psi\rangle = U_f(\alpha|0\rangle + \beta|1\rangle) \otimes |0\rangle = \alpha U_f|0\rangle|0\rangle + \beta U_f|1\rangle|0\rangle = \alpha|0\rangle|f(0)\rangle + \beta|1\rangle|f(1)\rangle.$$

Thus, we seem to have obtained both outputs of f with just a single query! Unfortunately, things in life are rarely free, and this is certainly no exception — although we have both outputs $f(0)$ and $f(1)$ in superposition, we cannot hope to *extract* both answers via measurement. In particular, once we measure both registers in the standard basis, we will collapse to one of the two terms in the superposition, effectively destroying the other term.

Exercise. Suppose one measures the first qubit of the output state $|\psi\rangle$ (this qubit marks which term in the superposition we have) with a standard basis measurement $\{|0\rangle\langle 0|, |1\rangle\langle 1|\}$. Show that the probability of outcome 0 or 1 is $|\alpha|^2$ or $|\beta|^2$, respectively, and that in each case, the state collapses to either $|0\rangle|f(0)\rangle$ or

$|1\rangle|f(1)\rangle$, respectively. Thus, only one answer $f(0)$ or $f(1)$ can be extracted this way.

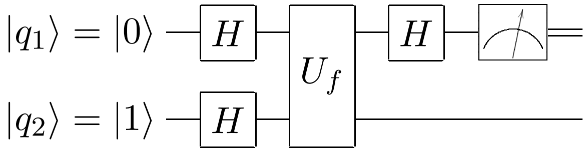
Luckily, our goal is *not* to extract both $f(0)$ and $f(1)$ after a single query. Rather, we want something possibly simpler: To evaluate the expression $f(0) \oplus f(1)$.

Exercise. Convince yourself that f is constant if $f(0) \oplus f(1) = 0$ and f is balanced if $f(0) \oplus f(1) = 1$. Thus, Deutsch's problem is equivalent to evaluating $f(0) \oplus f(1)$.

It turns out that by a clever twist of the naive approach above, we can indeed evaluate $f(0) \oplus f(1)$ (*without* individually obtaining the values $f(0)$, $f(1)$) via Deutsch's algorithm.

3.2 Deutsch's algorithm

The circuit for Deutsch's algoritm is given as follows.



It is not *a priori* obvious at all why this circuit should work, and this is indicative of designing quantum algorithms in general — the methods used are often incomparable to known classical algorithm design techniques, and thus developing an intuition for the quantum setting can be very difficult. Let us hence simply crunch the numbers and see why this circuit indeed computes $f(0) \oplus f(1)$, as claimed. Once we've done the brute force calculation, we will take a step back and talk about the *phase kickback* trick, which is being used here, and which will allow for a much simpler and somewhat more intuitive understanding of why the algorithm works.

As in previous lectures, let us divide the computation into 4 stages denoted by the quantum state in that stage: At the start of the circuit ($|\psi_1\rangle$), after the first Hadamards are applied ($|\psi_2\rangle$), after U_f is applied ($|\psi_3\rangle$), and after the last Hadamard is applied ($|\psi_4\rangle$). It is clear that

$$\begin{aligned} |\psi_1\rangle &= |0\rangle|1\rangle, \\ |\psi_2\rangle &= |+\rangle|-\rangle = \frac{1}{2}(|0\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|0\rangle - |1\rangle|1\rangle). \end{aligned}$$

After the oracle U_f is applied, we have state

$$|\psi_3\rangle = \frac{1}{2}(|0\rangle|f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|f(1)\rangle - |1\rangle|1 \oplus f(1)\rangle).$$

Before we apply the final Hadamard, it will be easier to break our analysis down into two cases: When f is constant and when f is balanced.

Case 1: Constant f. By definition, if f is constant, then $f(0) = f(1)$. Therefore, we can simplify $|\psi_3\rangle$ to

$$\begin{aligned} |\psi_3\rangle &= \frac{1}{2}(|0\rangle|f(0)\rangle - |0\rangle|1 \oplus f(0)\rangle + |1\rangle|f(0)\rangle - |1\rangle|1 \oplus f(0)\rangle) \\ &= \frac{1}{2}((|0\rangle + |1\rangle) \otimes |f(0)\rangle - (|0\rangle + |1\rangle) \otimes |1 \oplus f(0)\rangle) \\ &= \frac{1}{2}(|0\rangle + |1\rangle) \otimes (|f(0)\rangle - |1 \oplus f(0)\rangle) \\ &= \frac{1}{\sqrt{2}}|+\rangle \otimes (|f(0)\rangle - |1 \oplus f(0)\rangle). \end{aligned}$$

Thus, qubit 1 is now in state $|+\rangle$. We conclude that

$$|\psi_4\rangle = \frac{1}{\sqrt{2}}|0\rangle \otimes (|f(0)\rangle - |1 \oplus f(0)\rangle),$$

i.e. qubit 1 is exactly in state $|0\rangle$. Thus, measuring qubit 1 in the standard basis now yields outcome 0 with certainty.

Case 2: Balanced f. By definition, if f is balanced, then $f(0) \neq f(1)$. Since f is a binary function, this means $f(0) \oplus 1 = f(1)$ and equivalently $f(1) \oplus 1 = f(0)$. Therefore, we can simplify $|\psi_3\rangle$ to

$$\begin{aligned}
|\psi_3\rangle &= \frac{1}{2}(|0\rangle|f(0)\rangle - |0\rangle|f(1)\rangle + |1\rangle|f(1)\rangle - |1\rangle|f(0)\rangle) \\
&= \frac{1}{2}((|0\rangle - |1\rangle) \otimes |f(0)\rangle - (|0\rangle - |1\rangle) \otimes |f(1)\rangle) \\
&= \frac{1}{2}(|0\rangle - |1\rangle) \otimes (|f(0)\rangle - |f(1)\rangle) \\
&= \frac{1}{\sqrt{2}}|-\rangle \otimes (|f(0)\rangle - |f(1)\rangle).
\end{aligned}$$

Thus, qubit 1 is now in state $|-\rangle$. We conclude that

$$|\psi_4\rangle = \frac{1}{\sqrt{2}}|1\rangle \otimes (|f(0)\rangle - |f(1)\rangle),$$

i.e. qubit 1 is exactly in state $|1\rangle$. Thus, measuring qubit 1 in the standard basis now yields outcome 1 with certainty.

Conclusion. If f is constant, the algorithm outputs 0, and if f is balanced, the algorithm outputs 1. Thus, the algorithm decides whether f is constant or balanced, using just a single query!

3.3 The phase kickback trick

We've analyzed Deutsch's algorithm using a brute force calculation, but there's a more intuitive view which will be used repeatedly in later algorithms, and which simplifies our calculation here greatly. This view is in terms of the *phase kickback trick*, which Deutsch's algorithm uses. To explain the trick, consider for any $x \in \{0,1\}$ what happens if we run U_f on input $|x\rangle|-\rangle$:

$$|\psi\rangle = U_f|x\rangle|-\rangle = \frac{1}{\sqrt{2}}(U_f|x\rangle|0\rangle - U_f|x\rangle|1\rangle) = \frac{1}{\sqrt{2}}(|x\rangle|f(x)\rangle - |x\rangle|1 \oplus f(x)\rangle) = |x\rangle \otimes \frac{1}{\sqrt{2}}(|f(x)\rangle - |1 \oplus f(x)\rangle).$$

Now, there are two possibilities: Either $f(x) = 0$, or $f(x) = 1$. If $f(x) = 0$, the equation above simplifies to

$$|\psi\rangle = |x\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |x\rangle|-\rangle,$$

i.e. the input state is unchanged by the action of U_f. If, on the other hand, $f(x) = 1$, we instead have

$$|\psi\rangle = |x\rangle \otimes \frac{1}{\sqrt{2}}(|1\rangle - |0\rangle) = -|x\rangle|-\rangle,$$

i.e. a -1 phase factor is produced. We can summarize both these cases in a single equation:

$$U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle. \tag{1}$$

Exercise. Convince yourself that the equation above indeed captures both the cases of $f(x) = 0$ and $f(x) = 1$.

Reanalyzing Deutsch's algorithm using phase kickback. Let us return to the state in Deutsch's algorithm just before U_f is applied, i.e.

$$|\psi_2\rangle = |+\rangle|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle|-\rangle + |1\rangle|-\rangle).$$

(Note that we have not expanded out the $|-\rangle$ state as we did previously — this is because with the phase kickback trick, we don't need to go to this level of detail!) By applying phase kickback (Equation (1)), we know that after U_f is applied, we have state

$$|\psi_3\rangle = \frac{1}{\sqrt{2}}((-1)^{f(0)}|0\rangle|-\rangle + (-1)^{f(1)}|1\rangle|-\rangle).$$

Suppose now that f is constant, i.e. $f(0) = f(1)$. Then, above we can factor out the -1 phase factor to simplify $|\psi_3\rangle$ to

$$|\psi_3\rangle = (-1)^{f(0)}\frac{1}{\sqrt{2}}(|0\rangle|-\rangle + |1\rangle|-\rangle) = (-1)^{f(0)}|+\rangle|-\rangle.$$

Thus, applying the final Hadamard to qubit 1 yields

$$|\psi_4\rangle = (-1)^{f(0)}|0\rangle|-\rangle.$$

Measuring the first qubit now yields outcome 0 with certainty, as before.

On the other hand, if f is balanced (i.e. $f(0) \neq f(1)$), then we cannot simply factor out the (-1) term as before! Thus, up to an overall factor of ± 1, $|\psi_3\rangle$ can be written as

$$|\psi_3\rangle = \pm\frac{1}{\sqrt{2}}(|0\rangle|-\rangle - |1\rangle|-\rangle) = \pm|-\rangle|-\rangle.$$

Exercise. Verify the equation above by considering the two possible balanced functions $f_1(0) = 0$ and $f_1(1) = 1$ and $f_2(0) = 1$ and $f_2(1) = 0$.

We conclude that applying the final Hadamard to qubit 1 yields

$$|\psi_4\rangle = \pm|1\rangle|-\rangle.$$

Measuring the first qubit now yields outcome 1 with certainty, as before.

4 The Deutsch-Josza algorithm

Deutsch's algorithm works in the simple case where $f : \{0,1\} \mapsto \{0,1\}$ acts on a single input bit. However, single-bit functions are not so interesting; our primary area of interest is the design and analysis of quantum algorithms for determining properties of functions $f : \{0,1\}^n \mapsto \{0,1\}$ which act on *many* input bits. This requires some familiarity in handling n-qubit states, and a good way to practice this is by developing the n-bit generalization of Deutsch's algorithm, known as the Deutsch-Josza algorithm.

Specifically, imagine now we have an n-bit function $f : \{0,1\}^n \mapsto \{0,1\}$ which is promised to be constant or balanced, and we wish to determine which is the case. Here, constant means $f(x)$ is the same for all $x \in \{0,1\}$, and balanced means $f(x) = 0$ for precisely half the $x \in \{0,1\}^n$ and $f(x) = 1$ for the remaining inputs?

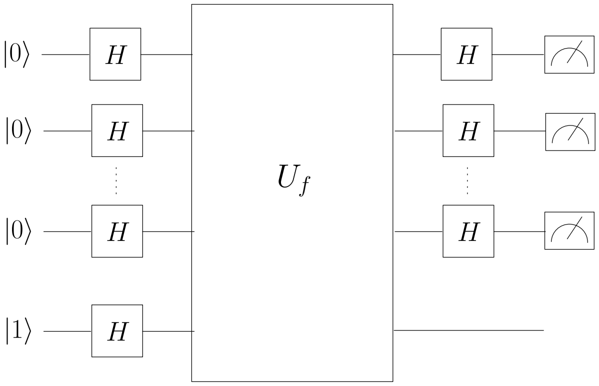


Figure 1: Quantum circuit for the Deutsch-Josza algorithm.

Exercise. Give examples of balanced and constant functions on 2 bits. Can you give an example of a 2-bit function which is *neither* constant nor balanced? Finally, can a single-bit function be neither constant nor balanced?

It turns out that Deutsch's algorithm generalizes in an easy manner to this setting; however, its analysis is a bit more tricky, and crucially uses the phase kickback trick. In this more general setting, note that we define the oracle U_f implementing f analogously to before: $U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$, where now x is an n-bit string.

The circuit for the Deutsch-Josza algorithm is given in Figure 4. As before, each wire denotes a single qubit. The first n qubits are initialized to $|0\rangle$; these are the input qubits. The final, i.e. $(n+1)$st, qubit is initialized to $|1\rangle$. Observe that the algorithm is the straightforward generalization of Deutsch's algorithm to the setting of n input qubits. We claim that using a single query to U_f, the Deutsch-Josza algorithm can determine if f is constant or balanced. Let us now see why this is so.

As before, we divide the computation into 4 stages denoted by the quantum state in that stage: At the start of the circuit ($|\psi_1\rangle$), after the first Hadamards are applied ($|\psi_2\rangle$), after U_f is applied ($|\psi_3\rangle$), and after the last Hadamard is applied ($|\psi_4\rangle$). It is clear that

$$\begin{aligned}|\psi_1\rangle &= |0\rangle\cdots|0\rangle|1\rangle = |0\rangle^{\otimes n}|1\rangle,\\ |\psi_2\rangle &= |+\rangle\cdots|+\rangle|-\rangle = |+\rangle^{\otimes n}|1\rangle.\end{aligned}$$

Since we have defined the action of U_f in terms of the standard basis, i.e. $U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$, in order to understand how U_f applies to $|\psi_2\rangle$, we first need to rewrite $|+\rangle^{\otimes n}$ in terms of the standard basis. For this, note that

$$|+\rangle^{\otimes n} = \frac{1}{\sqrt{2}^n}(|0\rangle + |1\rangle) \otimes \cdots \otimes (|0\rangle + |1\rangle) = \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle,$$

where the last equality holds since expanding the tensor products out yields 2^n terms in the sum, each of which corresponds to a unique string $x \in \{0,1\}^n$.

Exercise. Verify that $|+\rangle^{\otimes 3} = \frac{1}{2\sqrt{2}} \sum_{x\in\{0,1\}^3} |x\rangle$.

It follows that we can rewrite $|\psi_2\rangle$ as

$$|\psi_2\rangle = |+\rangle^{\otimes n}|1\rangle = \frac{1}{2^{n/2}} \sum_{x \in \{0,1\}^n} |x\rangle|-\rangle.$$

Now that we have the first register written with respect to the standard basis, we can analyze the action of U_f using the phase kickback trick, obtaining state

$$|\psi_3\rangle = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle|-\rangle.$$

Finally, we must apply the last set of Hadamard gates, which gives $|\psi_4\rangle = \frac{1}{2^{n/2}}\sum_{x\in\{0,1\}^n}(-1)^{f(x)}H^{\otimes n}|x\rangle|-\rangle$. To analyze this state, we first need to understand what $H^{\otimes n}|x\rangle$ equals for arbitrary $x \in \{0,1\}$. For this, we begin with a clean and formal way for writing the action of H on a *single* qubit. Recall that $H|0\rangle = |+\rangle$ and $H|1\rangle = |-\rangle$. Equivalently, this means that for $x_1 \in \{0,1\}$,

$$H|x_1\rangle = \frac{1}{\sqrt{2}} \sum_{z_1\in\{0,1\}} (-1)^{x_1 z_1}|z_1\rangle,$$

where $x_1 z_1$ is just the product of x_1 and z_1.

Exercise. Verify the statement above by considering the cases $H|0\rangle$ and $H|1\rangle$.

Now that we have a clean way of expressing $H|x_1\rangle$ for single qubit $|x_1\rangle$, we can generalize this to n-qubit states. Specifically, if we write string $x = x_1 \cdots x_n$ as $|x_1\rangle \otimes \cdots \otimes |x_n\rangle$, we have

$$\begin{aligned}
H^{\otimes n}|x\rangle &= H|x_1\rangle \otimes \cdots \otimes H|x_n\rangle \\
&= \frac{1}{2^{n/2}} \sum_{z_1\in\{0,1\}} (-1)^{x_1 z_1}|z_1\rangle \otimes \cdots \otimes \sum_{z_n\in\{0,1\}} (-1)^{x_n z_n}|z_n\rangle \\
&= \frac{1}{2^{n/2}} \sum_{z\in\{0,1\}^n} (-1)^{x_1 z_1 + \cdots + x_n z_n}|z\rangle.
\end{aligned}$$

Can we simplify this expression further? There is one small last trick we can apply which will clean it up a bit: Observe that $x_1z_1 + \cdots x_nz_n$ can be rewritten as the bitwise inner product modulo 2 of strings x and z, i.e. $x_1z_1 + \cdots x_nz_n = x \cdot z$. (The mod 2 arises since the base is (-1), so all we care about is if the exponent $x \cdot z$ is even or odd.) Combining these facts, we have that after the final Hadamards are applied, we have

$$\begin{aligned}
|\psi_4\rangle &= \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} H^{\otimes n}|x\rangle|-\rangle \\
&= \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)} \left(\frac{1}{2^{n/2}} \sum_{z\in\{0,1\}^n} (-1)^{x\cdot z}|z\rangle \right) |-\rangle \\
&= \frac{1}{2^n} \sum_{z\in\{0,1\}^n} \sum_{x\in\{0,1\}^n} (-1)^{f(x)+x\cdot z}|z\rangle|-\rangle. \qquad (2)
\end{aligned}$$

Finally, a measurement of the first n qubits of $|\psi_4\rangle$ is made in the standard basis. As for Deutsch's algorithm, a good way to analyze this is by individually considering the cases when f is constant or balanced, respectively. The trick in both analyses will be to determine the amplitude on the all zeroes state, $|0\rangle^{\otimes n}$, in the first register.

Case 1: Constant f. Suppose first that f is constant. Then, we can factor out the term $(-1)^{f(x)}$ in $|\psi_4\rangle$, i.e.

$$|\psi_4\rangle = (-1)^{f(x)} \sum_{z\in\{0,1\}^n} \left(\frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot z} \right) |z\rangle|-\rangle.$$

Now consider the amplitude on $|z\rangle = |0\cdots 0\rangle$, which is given by (up to an insignificant $(-1)^{f(x)}$ global phase out front)

$$\frac{1}{2^n}\sum_{x\in\{0,1\}^n}(-1)^{x\cdot 0\cdots 0} = \frac{1}{2^n}\sum_{x\in\{0,1\}^n}(-1)^0 = \frac{1}{2^n}\sum_{x\in\{0,1\}^n}1 = \frac{1}{2^n}2^n = 1.$$

In other words, the state $|0\rangle^{\otimes n}|-\rangle$ has amplitude 1. Since $|\psi_4\rangle$ is a unit vector, we conclude that we must have $|\psi_4\rangle = (-1)^{f(x)}|0\cdots 0\rangle|-\rangle$, i.e. all the weight is on this one term. Thus, if f is constant, then measuring the first n qubits in the standard basis yields outcome $|0\cdots 0\rangle$ with certainty.

Case 2: Balanced f. In this case, we cannot simply factor out the $(-1)^{f(x)}$ term, since f can take on different values depending on its input x. However, it still turns out to be fruitful to think about the amplitude on the state $|z\rangle = |0\cdots 0\rangle$. This is given by

$$\frac{1}{2^n}\sum_{x\in\{0,1\}^n}(-1)^{f(x)+x\cdot 0\cdots 0} = \frac{1}{2^n}\sum_{x\in\{0,1\}^n}(-1)^{f(x)}$$

since $z = 0$. Since f is balanced, we know that for precisely half the terms in this sum, $f(x) = 0$, and for the other half $f(x) = 1$. In other words, all the terms in this sum cancel, i.e. the sum equals 0! We conclude that the amplitude on $|z\rangle = |0\cdots 0\rangle$ is zero, and so we will never see outcome $|0\cdots 0\rangle$ in the final measurement.

Combining our observations for both cases, we find the following: When the final measurement is completed, if the outcome is 0^n, then we output "constant"; for any other n-bit measurement result, we output "balanced".

Classical algorithms for the Deutsch-Josza problem. Finally, you might wonder how *classical* algorithms compete with the Deutsch-Josza algorithm. For the case of *deterministic* classical algorithms, if f is balanced, then there are $2^{n/2}$ inputs x yielding $f(x) = 0$ and $2^{n/2}$ inputs x yielding $f(x) = 1$. Thus, in the worst case, an algorithm might be very unlucky and have its first $2^{n/2}$ queries return value $f(x) = 0$, only to have query $2^{n/2} + 1$ return $f(x) = 1$. For this reason, deterministic algorithms have worst case query complexity $2^{n/2} + 1$. In this setting, the Deutsch-Josza algorithm yields an exponential improvement over classical algorithms, requiring just a single query to f.

However, one can also try a *randomized* classical algorithm. Here is a particularly simple one:

1. Repeat the following K times, for K a parameter to be chosen as needed:
   (a) Pick $x \in \{0,1\}$ uniformly at random.
   (b) Call U_f to evaluate $f(x)$.
   (c) If $f(x)$ differs from any previous query answer, then halt and output "balanced".
2. Halt and output "Constant".

This algorithm does something straightforward — it repeatedly tries random values of $f(x)$, and if it ever obtains two different answers to its queries, it outputs "balanced"; otherwise, all its queries returned the same answer, and so it outputs "constant". Will this algorithm always be correct? *No.* In fact, it has *one-sided error*, in the following sense. If f is constant, then all queries will always output the same answer. Thus, line 1c will never cause the program to halt, and it will correctly output "constant" on line 2. On the other hand, if f is balanced, then the algorithm might get really unlucky — all of its K queries $f(x)$ might output the same bit, even though f is balanced. Thus, in this case the algorithm will incorrectly output "constant" on line 2.

We are left with two questions: What is the query cost of this randomized algorithm, and what is its probability of error? Clearly, the cost is K queries, since that is the number of times the loop runs. As for the error, note that when f is balanced (which is the only case in which an error might occur), when making a single query, the probability of getting output (say) $f(x) = 0$ is $1/2$. Since all query inputs are uniformly and independently chosen at random, the probability of having all K queries return the same bit is hence $1/2^{K-1}$ (the -1 in the exponent is because there are two such cases, strings 0^K and 1^K). We conclude that the error probability scales inverse exponentially with K.

Exercise. Suppose we wish our randomized algorithm to have error probability at most $1/n$. What should we set K to?

Finally, let us compare this to the Deutsch-Josza algorithm. Suppose that f is chosen to be constant with probability 1/2 and balanced with probability 1/2. Then, the Deutsch-Josza algorithm uses 1 query to determine if f is constant or balanced with certainty. On the other hand, if the randomized algorithm uses $K \in O(1)$ queries, then its probability of success is

$$\begin{aligned}
\Pr[\text{success}] &= \Pr[f \text{ is constant}] \cdot \Pr[\text{output ``constant''} \mid \text{f is constant}] + \\
&\quad \Pr[f \text{ is balanced}] \cdot \Pr[\text{output ``balanced''} \mid \text{f is balanced}] \\
&= \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot (1 - \frac{1}{2^{K-1}}) \\
&= 1 - \frac{1}{2^K}.
\end{aligned}$$

Exercise. What is the probability of success for the randomized algorithm if it performs just a single query, i.e. $K = 1$? How does this compare with the Deutsch-Josza algorithm?

5 The Berstein-Vazirani algorithm

Although interesting as a proof of principle, the Deutsch-Josza algorithm does not solve a particularly motivated problem. However, as the second quote opening this lecture note suggests[1], "where there's smoke, there's fire", and a closer look at the Deutsch-Josza algorithm will reveal something more interesting at play.

To begin, recall Equation (2), which stated that just before measuring in the standard basis, the Deutsch-Josza algorithm yields state

$$|\psi_4\rangle = \frac{1}{2^n} \sum_{z \in \{0,1\}^n} \left(\sum_{x \in \{0,1\}^n} (-1)^{f(x) + x \cdot z} \right) |z\rangle|-\rangle. \tag{3}$$

Note that this derivation was *independent* of the definition of f. This raises the question: For what choices of f could measuring $|\psi_4\rangle$ yield interesting information? Since the exponent on -1 contains $x \cdot z$, a naive guess might be to consider linear functions $f(x) = a \cdot x$, where $a, x \in \{0,1\}^n$ (or more generally affine functions $f(x) = a \cdot x + b$ for $b \in \{0,1\}$). This seems a natural guess, as it would allow us to factor out the a term. Specifically, plugging $f(x) = a \cdot x$ into Equation (3) yields

$$|\psi_4\rangle = \frac{1}{2^n} \sum_{z \in \{0,1\}^n} \left(\sum_{x \in \{0,1\}^n} (-1)^{(a+z) \cdot x} \right) |z\rangle|-\rangle. \tag{4}$$

Thus, the probability of obtaining some outcome z when measuring the first register in the standard basis is $(2^{-n} \sum_{x \in \{0,1\}^n} (-1)^{(a+z) \cdot x})^2$. Since our goal is to extract a, let us ask: What is the probability of observing $z = a$? This is given by (recall we are working modulo 2 in the exponent)

$$\left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{(a+a) \cdot x} \right)^2 = \left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} (-1)^{0 \cdot x} \right)^2 = \left(\frac{1}{2^n} \sum_{x \in \{0,1\}^n} 1 \right)^2 = 1. \tag{5}$$

In other words, all of the amplitude in $|\psi_4\rangle$ is concentrated on the $|a\rangle$ term. Equivalently, $|\psi_4\rangle = |a\rangle|-\rangle$. We thus have that measuring the first register yields a with certainty. This procedure is known as the Bernstein-Vazirani algorithm, which solves the problem: Given a black-box linear function $f(x) = a \cdot x$, determine a.

[1] We are slightly abusing this proverb here, in that it typically has a negative connotation, whereas here our premise is that quantum computation outperforming classical computation is a good thing.

Exercise. Is it possible to have $z \neq a$ in Equation (4)? In other words, does there exist a term in the superposition with non-zero amplitude which satisfies $z \neq a$?

Exercise. Given an affine function $f(x) = ax + b$, show how the Bernstein-Vazirani algorithm allows one to extract a with a single quantum query.

Interpretation via the Deutsch-Josza problem. To close the lecture, let us go full circle and see what Bernstein-Vazirani teaches us about Deutsch-Josza. Suppose first that $f(x) = b$ for $b \in \{0, 1\}$, i.e. we have affine function $f(x) = ax + b$ with $a = 0$ in the Bernstein-Vazirani framework. Then, clearly f is constant in the Deutsch-Josza framework. In both algorithms (which are the same algorithm), we hence obtain all-zeroes in the first register when we measure $|\psi_4\rangle$.

Next, consider $f(x) = ax + b$ for $a \neq 0$. (By an exercise above, we can ignore b, as it simply leads to a global phase in $|\psi_4\rangle$, which cannot be observed.) Since the case of $a = 0$ corresponded to constant f, we now expect the $a \neq 0$ case to correspond to *balanced* f. The following basic, but important, fact in Boolean arithmetic confirms this, and is worth keeping in your toolkit moving forward.

Exercise. Let $f(x) = a \cdot x + b \mod 2$ be an affine function with $a \neq 0$. Then, for precisely half the inputs $x \in \{0, 1\}^n$, $f(x) = 0$, and on the other half of the inputs, $f(x) = 1$. In other words, f is balanced.