Notes for EECS 498-001

Quantum Programming

Albon Wu

April 6, 2024

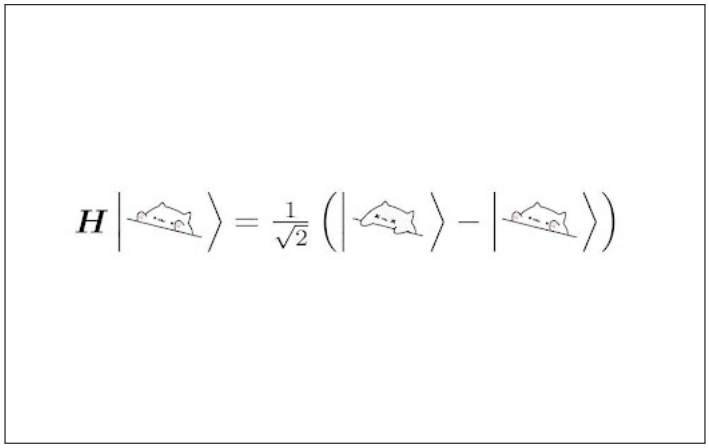
Contents

0 Introduction

Quantum circuits, algorithms, and computation.

Professor: Jon Beaumont.

Textbook: None.

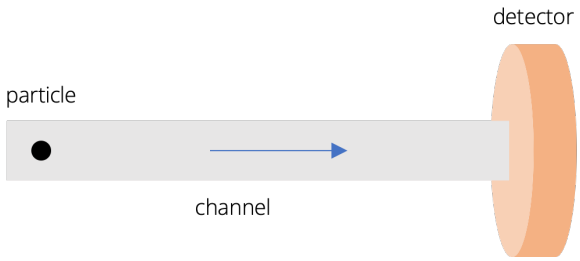


1 Quantum Logic

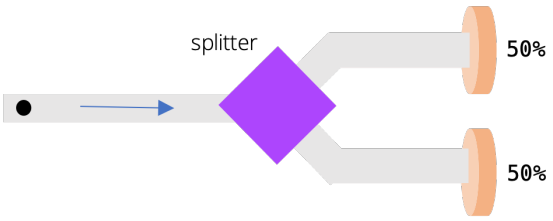
1.1 The Basics

Quantum computers use the laws of quantum mechanics as the basis for computation. Here's an experiment:

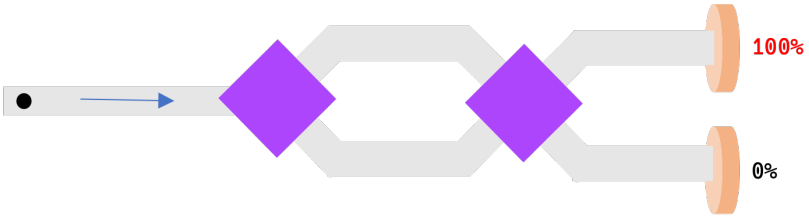
1. Take a single particle, like an electron
2. Isolate the particle so that nothing, not even light, interacts with it
3. Shoot it down a channel until it hits a detector



The expected behavior occurs; the detector observes the particle when it reaches the end of the channel. Now we introduce a splitter that has a 50/50 chance of deflecting the particle up or down.



As expected, each detector activates half of the time. But what if we merge the branches with another splitter?



We should see the same result, but in fact the top detector activates 100% of the time. This is because the particle does not take the top *or* bottom path; two different versions of the particle simultaneously take both paths.

That is, the particle is in **superposition** until it is observed. In the experiment, these superposed versions interfered with one another, canceling out the path to the bottom detector.

This is useful for computing because we can represent data using quantum bits, or **qubits**, which are bits in superposition of 0 and 1. A qubit can be 0, 1, or both, which lets us simultaneously represent every n-bit value using n qubits.

Incidentally, the superposition collapses when we observe it, so an effective quantum algorithm must take advantage of interference.

1.2 Math and Notation

To represent the probability of a quantum state, we use complex numbers. This lets us account for the cancellation of probabilities during interference.

The **magnitude** of a complex number is its distance from 0. Its **phase** is the angle formed by the positive x-axis in the complex plane. Addition and multiplication are defined intuitively and have the expected properties.

We can write $e^{ix} = \cos(x) + i\sin(x)$, which generalizes to the **polar form** of a complex number $re^{i\phi}$ with magnitude r and phase ϕ. This makes multiplication easier.

This is enough for us to express probabilities of quantum states. To express the states themselves we use **ket notation**, which looks like $|\psi\rangle$.

Consider an experiment with two possible outcomes: 0 or 1. We write $|0\rangle$ and $|1\rangle$ for the states where we definitely measure a 0 and 1, respectively. To represent superposition, the current state is written as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ for $\alpha, \beta \in \mathbb{C}$. α and β are called **probability amplitudes**.

$\alpha^*\alpha$ is then the probability that the state collapses to $|0\rangle$ upon measurement. We say that $|\psi\rangle$ represents the state of a **quantum bit**, or a **qubit**. Note that two qubits can have equal state probability distributions but different phases, which is useful for abstracting interference.

There is one caveat: states that differ only by a common phase factor—**global phase**—are considered physically equivalent. Global phase affects neither:

- state probability (only the magnitude of the probability amplitude does)
- nor interference (only relative phase does),

so it doesn't affect quantum state.

That is,

$$e^{i\theta}(|0\rangle + |1\rangle) \approx |0\rangle + |1\rangle$$
$$e^{i\theta}|0\rangle + |1\rangle \neq |0\rangle + |1\rangle.$$

1.3 Qubits

A quantum state can be described as a vector in a complex valued, N-dimensional vector space. This means that we can express a quantum state as a linear combination of N basis vectors (or **state vectors**), where each state vector is a potential outcome:

$$|\psi\rangle = \sum_j \alpha_j |j\rangle.$$

As before, the probability amplitudes α_j are complex. It's natural to then ask about the minimum amount of space needed to encode a qubit. For simplicity, consider a qubit $\alpha_0|0\rangle + \alpha_1|1\rangle$ in a 2D vector space.

In this example, four parameters definitely suffice: we can take the four total arguments of α_0 and α_1. In polar form, these are r_0, r_1, ϕ_0, ϕ_1.

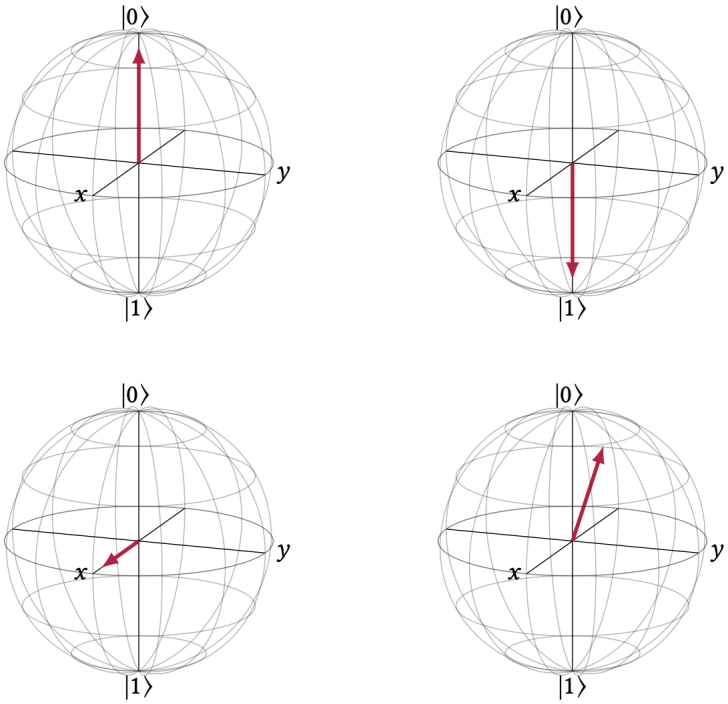
But since global phase is irrelevant, we can send the phase of one probability amplitude to 0 and adjust the other to preserve relative phase. Note also that the magnitudes (i.e., probabilities)

must add to one. So we can write the qubit's state as such:

$$|\psi\rangle = \cos(\theta/2)\,|0\rangle + (e^{i\phi})\sin(\theta/2)\,|1\rangle$$

using the well-known identity $\sin^2(\theta) + \cos^2(\theta) = 1$.

We can then graph θ and ϕ as coordinates on a **Bloch sphere**.



θ and ϕ play exactly the same role here as they do in spherical coordinates; θ is the angle from the z-axis and ϕ is the angle from the x-axis.

Intuitively, "latitude," or the height of the terminal point, represents the probability distribution between states $|0\rangle$ and $|1\rangle$. "Longitude" represents phase.

1.4 Single-Qubit Transformations

Take a general state vector $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$. The operation that swaps α and β is called an X-**transformation**:

$$X\,|\psi\rangle = \beta\,|0\rangle + \alpha\,|1\rangle\,.$$

This has the effect of rotating the Bloch sphere through the plane perpendicular to the x-axis. If we encode $|\psi\rangle$ by $[\alpha, \beta]^\top$, we can represent an X-transformation as follows:

$$X\,|\psi\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix}.$$

Note that X is given by the familiar linear transformation whose matrix swaps vector entries in $\mathbb{R}^2$.

We might then ask what other (linear) transformations exist for qubits. Importantly, since the probabilities in a qubit must add to 1, the matrix of a valid transformation must preserve length (the norm given by the space's inner product).

The matrices that yield length-preserving linear transformations are called **unitary operators**. A matrix M is unitary if it satisfies:

$$M^\dagger M = I$$
$$\iff \begin{bmatrix} a^* & c^* \\ b^* & d^* \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Since they do not scale points in space, unitary operators correspond to rotations of the Bloch sphere. Also, note that every unitary matrix M necessarily has an inverse $M^\dagger$.

Now that we know how to characterize valid qubit transformations, let's look for one that allows us to map $|0\rangle$ or $|1\rangle$ to a state not on the z-axis of the Bloch sphere. The X-transformation is insufficient since it simply inverts the state like a classical NOT gate.

Denote $|0\rangle$ by $[1, 0]^\top$. At first glance, it seems like we can just do the following:

$$\left(\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \right) \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}.$$

Unfortunately, this choice of matrix also gives

$$\left(\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \right) \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

and so the transformation is not injective; hence, the matrix is neither invertible nor unitary. But we can add a phase to $|1\rangle$ by negating its probability amplitude so that our matrix becomes

$$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

On the basis states, this transform yields

$$\left(\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \right) \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} =: |+\rangle$$
$$\left(\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \right) \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} =: |-\rangle$$

We call this the **Hadamard** or H**-operation**.

From before, we know that every unitary transformation preserves norm, meaning that they effectively rotate the Bloch sphere. H does a 180° rotation about the axis $x = z$.

Here's some intuition for why the rotation is 180°; grokking the axis is left as an exercise. We know already that $H|0\rangle = |+\rangle$. So computing $H^2|0\rangle = H|+\rangle$ yields, by linearity of H:

$$\begin{aligned} H|+\rangle &= H\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \right) \\ &= \frac{1}{\sqrt{2}} H|0\rangle + \frac{1}{\sqrt{2}} H|1\rangle \\ &= \frac{1}{2}(|0\rangle + |1\rangle) + \frac{1}{2}(|0\rangle - |1\rangle) \\ &= \frac{1}{2}|0\rangle + \frac{1}{2}|0\rangle \\ &= |0\rangle \end{aligned}$$

So applying the H-operation to the basis vector $|0\rangle$ twice yields itself, which we would expect from a 180° rotation.

But this looks familiar! This is exactly the outcome of our particle experiment from earlier, where a particle in superposition of two states ended up definitively in one of them due to interference.

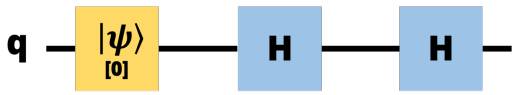
Now that we have two transformations, we can represent them jointly in a **circuit diagram**:

This is equivalent to the equation

$$\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix},$$

which represents a qubit initialized to $|\psi\rangle = |0\rangle$ that undergoes an X transformation followed by an H transformation. Each operator is shown as a gate in the circuit diagram. This representation is only conceptual; actual quantum computers suspend particles in a field and change their energies without physically moving them through gates.

We can also express our quantum experiment from earlier as follows:



Each splitter is functionally an H-gate that sends the particle into superposition of the basis states.

Some other gates include the Y-**gate**:

$$Y = \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix}.$$

This functions like X but adds an additional phase. We also have the Z-**gate**:

$$Z = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}.$$

This rotates the sphere 180° around the z-axis. The X, Y, and Z gates collectively are called **Pauli gates**.

There are a few more gates that operate similarly to Z but with more granular angles:

$$S = \begin{bmatrix}1 & 0\\ 0 & e^{\frac{i\pi}{2}}\end{bmatrix}.$$

$$T = \begin{bmatrix}1 & 0\\ 0 & e^{\frac{i\pi}{4}}\end{bmatrix}.$$

$$R_Z(\phi) = \begin{bmatrix} e^{\frac{-i\phi}{2}} & 0 \\ 0 & e^{\frac{i\phi}{2}} \end{bmatrix}.$$

Note that $S^2 = Z$ and $T^4 = Z$. $R_Z(\phi)$ rotates about the z-axis by ϕ.

It turns out that any quantum gate can be approximated with a combination of **basis gates**. One basis is $\{H, S, T\}$; in this class, we use $\{H, S, R_Z\}$.

1.5 Multi-Qubits

Multi-qubits encode states whose outcomes are multiple bits. So a 2-qubit state would look like the following:

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle.$$

We have four basis states, meaning that $|\psi\rangle$ is in a 4D complex vector space. The probability of measuring $|00\rangle$ is $|\alpha_{00}|^2$.

As before, we can drop two parameters from our representation (one for global phase, one accounting for total probability), so we need six parameters to encode $|\psi\rangle$. In general, the number of parameters is exponential in the number of qubits.

To describe multi-qubit states mathematically, we use the **tensor product** of multiple isolated qubits. In the two-qubit case, we write:

$$|ab\rangle = |a\rangle \otimes |b\rangle = \begin{bmatrix} a_0 \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \\ a_1 \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} a_0 b_0 \\ a_0 b_1 \\ a_1 b_0 \\ a_1 b_1 \end{bmatrix}.$$

We define the tensor product of matrices analogously. Consider matrices

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad N = \begin{bmatrix} e & f \\ g & h \end{bmatrix}.$$

Then $M \otimes N$ is given by:

$$M \otimes N = \begin{bmatrix} aN & bN \\ cN & dN \end{bmatrix} = \begin{bmatrix} ae & af & be & bf \\ ag & ah & bg & bh \\ ce & cf & de & df \\ cg & ch & dg & dh \end{bmatrix}.$$

This is useful because it lets us define gates for multi-qubits. For instance, we can write $X|q_1\rangle \otimes H|q_0\rangle = (X \otimes H)|q_1 q_0\rangle$.

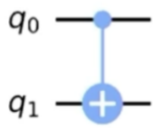
So if we want a multi-qubit gate that only acts on, say, the first qubit, we take the tensor product of some gate with the identity. The 2-qubit gate that only applies X to the first qubit is then given by $(X \otimes I)$ since $(X \otimes I)|q_1 q_0\rangle = X|q_1\rangle \otimes I|q_0\rangle = X|q_1\rangle \otimes q_0$.

But this single-qubit operation is kind of trivial. We would really benefit from gates where the qubits are not changing independently—gates like AND and OR.

Note that a traditional XOR gate fails, though, since it is not injective. We work around this by creating a gate with two outputs: for any inputs A and B, it outputs A and $A \wedge B$. This is called the **CNOT**, or **conditional-NOT gate**. Its truth table is as follows:

Input (t, c)	Output (t, c)
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|11\rangle$
$\|10\rangle$	$\|10\rangle$
$\|11\rangle$	$\|01\rangle$

We call t the target bit and c the control bit; c is always unchanged and t is overwritten with the XOR result. In a circuit diagram, the bits are drawn as follows:



The control qubit is filled and the target is drawn with a plus. By convention, in this class, the bottom bit (q_1 in this case) is taken to the be the most significant bit.

Example 1.5.1. Compute CNOT $|01\rangle$ given the configuration above.

Solution. In the diagram, q_1, the most significant bit, is the target. So we replace the leftmost qubit in $|01\rangle$ with $0 \wedge 1 = 1$ and preserve the control q_0, which is the least significant bit.

This gives CNOT $|01\rangle = |11\rangle$. □

We can also apply CNOT on more complex qubits.

Example 1.5.2. Compute CNOT $|0+\rangle$.

Solution. Note $|0+\rangle = |0\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |01\rangle)$. For the sake of notation, we will often omit factors of $\frac{1}{\sqrt{n}}$ from state vectors since we know they must be normalized. So $\frac{1}{\sqrt{2}}(|00\rangle + |01\rangle)$ is written as $|00\rangle + |01\rangle$.

In practice, we would obtain this multi-qubit state by applying CNOT to two qubits: both initialized to $|0\rangle$, and one passing through an H-gate.

We compute CNOT $|0+\rangle$ by applying CNOT to each basis state and combining the results using linearity, which gives CNOT $|0+\rangle$ = CNOT $|00\rangle$ + CNOT $|01\rangle$ = $|00\rangle + |11\rangle$. This is called the **Bell state**.

□

The Bell state has an interesting property: it is not equal to the tensor product of any two independent qubits. This is immediate from the fact that $|a\rangle \otimes |b\rangle = [a_0 b_0, a_0 b_1, a_1 b_0, a_1 b_1]^\top = [1, 0, 0, 1]^\top$ implies that $a_0 = 0$ or $b_1 = 0$, which is impossible since $a_0 b_0 = a_1 b_1 = 1$.

So $|00\rangle + |11\rangle$ cannot be factored into a product of single-qubit states; it is not a **product state**. Instead, it is an **entangled state**, which implies correlation between its constituent states. That is, one qubit is fully determined by the other; the value we measure for one qubit is the value for the other.

Returning to product states, consider the following example:

Example 1.5.3. Compute CNOT $|{-}{+}\rangle$ where the most significant bit is the target.

Solution. We start by writing $|{-}{+}\rangle = (|0\rangle - |1\rangle) \otimes (|0\rangle + |1\rangle) = |00\rangle - |10\rangle + |01\rangle - |11\rangle$. By direct computation, we get

$$\text{CNOT}(|00\rangle - |10\rangle + |01\rangle - |11\rangle) = |00\rangle - |10\rangle + |11\rangle - |01\rangle\,.$$

Note that this expression resembles $a^2 - ab - ba + b^2 = (a-b)^2$. Indeed, we can factorize it as $(|0\rangle - |1\rangle) \otimes (|0\rangle - |1\rangle) = |-\rangle|-\rangle = |{-}{-}\rangle$. □

That looks illegal. We just inverted the phase of the control! This phenomenon is known as **phase kickback**.

Why does this happen? Recall that CNOT is essentially a controlled X gate; we apply an X gate to the target iff the control bit is $|1\rangle$. But, like many linear transformations, X has eigenvectors (here, eigenstates) and eigenvalues (eigenphases).

For instance, $X|+\rangle = X(|0\rangle + |1\rangle) = |1\rangle + |0\rangle = 1\,|+\rangle$, so $|+\rangle$ is an eigenstate of X with eigenvalue 1. Similarly, $|-\rangle$ is an eigenstate of X with eigenvalue -1. Note that since our gates are unitary, any eigenvalue must be a magnitude-1 complex number; that is, of the form $e^{i\theta}$.

Multiplication by $e^{i\theta}$ corresponds to a phase addition of θ, which is consequential for controlled operations. Consider a controlled-U, or CU operation, where U is applied to a target qubit iff the control is 1. If $|\lambda\rangle$ is an eigenstate of U with eigenvalue $e^{i\theta}$, we have the following:

$$CU\,|\lambda{+}\rangle = CU(|\lambda\rangle|0\rangle + |\lambda\rangle|1\rangle) = |\lambda\rangle|0\rangle + e^{i\theta}\,|\lambda\rangle|1\rangle$$

where the last equality follows from the fact that the target bit is 1 in $|\lambda\rangle|1\rangle$ but not $|\lambda\rangle|0\rangle$. Since CU is controlled, the phase shift is only applied to one basis state, changing the relative phase.

2 Quantum Algorithms

2.1 Deutsch-Jozsa

As we alluded to earlier, the idea of quantum algorithms is to process a superposition of all possible inputs and orchestrate interference in a way that favors the correct output. The Deutsch-Jozsa algorithm uses a similar principle.

Consider the following question:

> **Question.** Take a function $f : \{0,1\} \to \{0,1\}$. Call f **balanced** if it outputs 0 for exactly half of its inputs and 1 for the other half. Otherwise, call it **constant**. Given an arbitrary f, is it balanced or constant?

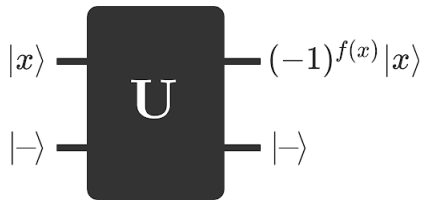
We call this the **Deutsch problem**.

It is easy to see that there are four $f : \{0,1\} \to \{0,1\}$ and that two of them are balanced and two are constant. A classical algorithm requires at least two computations of f, since computing $f(0)$ does not *a priori* tell us anything about $f(1)$, and vice versa.

Here, we will look at a quantum algorithm that solves the Deutsch problem using just one query of f.

First, it is worth asking how we even implement classical functions in quantum circuits. Take $f : \{0,1\} \to \{0,1\}$ such that $f(0) = f(1) = 1$. f is not reversible, so we can't implement it with unitary gates alone.

We resolve this by constructing a gate called a **quantum oracle** that sends an input $|yx\rangle$ to $|y \oplus f(x)\rangle\,|x\rangle$:



For simplicity, we will accept without proof that the quantum oracle is unitary. We call the top qubit the "data register" and the bottom one the "target register."

Now consider what happens when we initialize the target register to $|0\rangle$. If $f(x) = 1$, it is sent to $0 \oplus 1 = 1$. If $f(x) = 0$, it remains at $0 \oplus 0 = 0$. So setting the target register to $|0\rangle$ causes it to be inverted iff $f(x) = 1$; we call this a bitflip oracle.

It's then tempting to set the data register to $|+\rangle$, which sends the target register to $|f(0)\rangle\,|0\rangle + |f(1)\rangle\,|1\rangle$. This evaluates $f(0)$ and $f(1)$ simultaneously, but a measurement of the resulting state collapses into either $|f(0)\rangle\,|0\rangle$ *or* $|f(1)\rangle\,|1\rangle$. That is, we gain the same amount of information that a single query of f would give.

But the idea of initializing a register to a qubit in superposition is still reasonable. Let's try $|-x\rangle$, which sends the target to $|f(x)\rangle\,|x\rangle - |f(x) \oplus 1\rangle\,|x\rangle$.

If $f(x) = 0$, the output is $|0\rangle\,|x\rangle - |1\rangle\,|x\rangle$. Otherwise, it is $|1\rangle\,|x\rangle - |0\rangle\,|x\rangle$. We write $U_f\,|-x\rangle = (-1)^{f(x)}\,|-x\rangle$.

So we can say that setting the target register to $|-\rangle$ leaves the target register unchanged but flips the phase of the data register if $f(x) = 1$. This is phase kickback! We can take advantage of it by initializing the data register to a state in superposition.

Concretely, let's compute $U|-+\rangle$.

$$\begin{aligned} U|-+\rangle &= U(|-0\rangle + |-1\rangle) \\ &= (-1)^{f(0)}|-0\rangle + (-1)^{f(1)}|-1\rangle . \end{aligned}$$

Note that $|-0\rangle$ and $|-1\rangle$ have the same phase iff $f(0) = f(1)$:

$$(-1)^{f(0)}|-0\rangle + (-1)^{f(1)}|-1\rangle = \begin{cases} |-0\rangle + |-1\rangle = |-+\rangle & \text{if } f(0) = f(1) \\ |-0\rangle - |-1\rangle = |--\rangle & \text{if } f(0) \neq f(1). \end{cases}$$

So the data register is $|+\rangle$ if f is constant and $|-\rangle$ if it is balanced. These states are orthogonal; if we apply an H-gate and perform a measurement, we are guaranteed to distinguish them.

This is the **Deutsch algorithm**, and it offers a 2× speedup over classical algorithms for solving the Deutsch problem.

Now let's generalize to any function $f : \{0,1\}^n \longrightarrow \{0,1\}$. A precise statement of the problem is as follows:

> **Question.** Take a function $f : \{0,1\}^n \longrightarrow \{0,1\}$. Call f **balanced** if it outputs 0 for exactly half of its inputs and 1 for the other half. Call it **constant** if it outputs only 0 or only 1 for all its inputs. Given an arbitrary f, is it balanced or constant?

The strategy is very similar to Deutsch's algorithm, except we use multiple qubits instead of one $|x\rangle$.